

3-2015

Dense Point-Cloud Representation of a Scene using Monocular Vision

Yakov Diskin
University of Dayton

Vijayan K. Asari
University of Dayton, vasari1@udayton.edu

Follow this and additional works at: https://ecommons.udayton.edu/ece_fac_pub

 Part of the [Computer and Systems Architecture Commons](#), [Electrical and Electronics Commons](#),
and the [Systems and Communications Commons](#)

eCommons Citation

Diskin, Yakov and Asari, Vijayan K., "Dense Point-Cloud Representation of a Scene using Monocular Vision" (2015). *Electrical and Computer Engineering Faculty Publications*. 389.
https://ecommons.udayton.edu/ece_fac_pub/389

This Article is brought to you for free and open access by the Department of Electrical and Computer Engineering at eCommons. It has been accepted for inclusion in Electrical and Computer Engineering Faculty Publications by an authorized administrator of eCommons. For more information, please contact frice1@udayton.edu, mschlangen1@udayton.edu.

Journal of Electronic Imaging

JElectronicImaging.org

Dense point-cloud representation of a scene using monocular vision

Yakov Diskin
Vijayan Asari

Dense point-cloud representation of a scene using monocular vision

Yakov Diskin* and Vijayan Asari

University of Dayton, 300 College Park, Dayton, Ohio 45469-0232, United States

Abstract. We present a three-dimensional (3-D) reconstruction system designed to support various autonomous navigation applications. The system presented focuses on the 3-D reconstruction of a scene using only a single moving camera. Utilizing video frames captured at different points in time allows us to determine the depths of a scene. In this way, the system can be used to construct a point-cloud model of its unknown surroundings. We present the step-by-step methodology and analysis used in developing the 3-D reconstruction technique. We present a reconstruction framework that generates a primitive point cloud, which is computed based on feature matching and depth triangulation analysis. To populate the reconstruction, we utilized optical flow features to create an extremely dense representation model. With the third algorithmic modification, we introduce the addition of the preprocessing step of nonlinear single-image super resolution. With this addition, the depth accuracy of the point cloud, which relies on precise disparity measurement, has significantly increased. Our final contribution is an additional postprocessing step designed to filter noise points and mismatched features unveiling the complete dense point-cloud representation (DPR) technique. We measure the success of DPR by evaluating the visual appeal, density, accuracy, and computational expense and compare with two state-of-the-art techniques. © 2015 SPIE and IS&T [DOI: [10.1117/1.JEI.24.2.023003](https://doi.org/10.1117/1.JEI.24.2.023003)]

Keywords: three-dimensional reconstruction; dense point-cloud representation; depth resolution enhancement; super resolution; aerial surveillance; monocular vision; structure from motion.

Paper 14336P received Jun. 6, 2014; accepted for publication Feb. 5, 2015; published online Mar. 6, 2015.

1 Introduction

The challenge of replicating the human ability to analyze a scene or environment has recently been put on the forefront. Humans have an extraordinary ability to recognize objects and determine their depth in a variety of lighting conditions and at relatively large distances. Devices and sensors, such as traditional televisions and modern digital CCD cameras, attempt to simulate human vision by outputting projective images. As these technologies continue to develop, automation algorithms for scene understanding, which incorporate various viewing angles, time information, and complex geometries, will continue pursuing the task of rendering three-dimensional (3-D) models with human-like accuracy and computational speed. The aim of this research is to aid in scene depth understanding for autonomous machines, such as unmanned aerial vehicles and unmanned ground/surface vehicles. In order for an unmanned aerial system (UAS) to be able to navigate itself through an unknown environment, a real-time procedure analyzing the scene must be developed. Other techniques have already been attempted and somewhat successfully conducted using systems with multiple imaging sensors and range finders.¹⁻⁵ In this paper, our objective is to utilize a minimal number of sensors to construct a model of a scene. All of our experiments are conducted using only commercially available electro-optic cameras, such as a digital SLR camera. In this paper, all analysis is computed from the imagery; more specifically, we have constrained ourselves from utilizing depth sensors, global positioning system, or orientation devices.

We present a fully automatic technique, namely the dense point-cloud representation (DPR), capable of generating a 3-D scene of an unknown environment using only a single high-resolution moving camera. The algorithm generates a model obtained by correctly positioning millions of points into a 3-D Cartesian coordinate system. Each point in the model corresponds to a feature point in the real-world (RW) environment. As the camera travels through a scene, the same feature points are seen from a variety of locations. The features are tracked from frame to frame as they undergo changes in orientation and scale. The image features can also be used to estimate the camera position and orientation within an unknown environment. By determining the camera position and orientation corresponding to every frame, we are able to perform triangulation to compute the (x, y, z) coordinates for each point in the scene. This feature tracking and triangulation concept serves as the basis of the scene reconstruction algorithm.

The paper is structured as follows. In Sec. 2, we describe the current state-of-the-art reconstruction techniques, visual structure from motion (VSFM) and probabilistic volumetric representation (PVR). These techniques have been used in a multitude of applications and have a proven record in building appealing models using certain datasets. Next, in Sec. 3, we present the algorithmic framework for depth computation from uncalibrated monocular vision. We perform an evaluation of the framework which identifies the aspects in the algorithm that need further enhancements. In Sec. 4, we improve the algorithm by increasing the number of features used and, as a result, create an extremely dense 3-D point-cloud model. Once again, we provide an evaluation

*Address all correspondence to: Yakov Diskin, E-mail: diskiny1@udayton.edu

of the resulting 3-D models and present a way to increase the precision within the model. In Sec. 5, we present the technology and results generated using a novel single-image super-resolution technique, which allows us to enhance the depth resolution of the scene. In Sec. 6, we conclude the presentation of the full DPR technique by presenting an adaptive noise suppression technique that cleans the 3-D model and increases the model's usability. In Sec. 7, we perform a comprehensive comparison between our method and the state-of-the-art methods. We evaluate the techniques based on visual appeal, density, computational expense, and usability. Metrics, figures, and tables highlight the aspects in which each technique outperforms the others. Finally, we provide conclusions and an algorithmic summary in Sec. 8.

2 Background and Related Work

Monocular vision, better described as vision through a single camera source, presents new challenges when compared to stereo vision or a multicamera system. In a stereo system, similar to human vision, distances between cameras (the baseline) and their orientation is known and, in most circumstances, remain constant. In order to generate various viewing angles with a monocular system, the camera must continuously be moving. With a moving camera, the system obtains two different viewing angles from two points in time. The challenge is then to accurately compute the distance the camera has traveled or the exact location of the camera at each frame of video. In addition, the orientation of the camera at each point in time must be computed from the scene.

The input imagery can be affected by a variety of components. It is important to understand the challenges associated with the inputs and their potential effect on the resulting reconstruction. Variables, such as the electro-optic sensors, image resolution, scene contrast, exposure time, and blurriness, all add to the complexity of analyzing a scene and processing the imagery. The research field of automatic 3-D reconstruction of a scene is relatively young and, therefore, contains no standardized datasets for all to use. Within the literature, each 3-D reconstruction technique collects its own data and produces models in its own format. To cover the span of input scenarios, throughout this paper, we present a variety of data ranging in various ground sampling distances (GSD), lighting environments, frame rates, stability, and speed of sensor platform. Our featured datasets can be categorized as (1) ideal laboratory conditions, (2) captures at low altitude or ground level (0 to 15 feet), (3) aerial capture from mid-altitude (15 to 3000 feet), and (4) extremely high altitude (3000+ feet). A laboratory environment allows us to capture data in ideal conditions, where the lighting is uniformly distributed over the objects of interest and the camera is traveling with known speed and known orientation in relation to the scene. The objects, such as colorful water bottles, flags, and faces, were chosen to test the colorization capabilities and the depth computations of the reconstruction framework. Low-altitude or ground-level captures correspond to imagery taken by an unmanned ground vehicle (UGV). These data will contain variations and instability from an inconsistent speed and road unevenness. Mid-altitude captures tend to be extremely unstable sequences that make feature matching extremely complicated. During the data collection for this research, we have tested aerial scene capture over Providence (Rhode Island), Dayton

(Ohio), Columbus (Ohio), and Gary (Indiana). Aerial imagery is challenging due to the constant alterations in depth and scene resolution, which are dependent on the GSD that rapidly changes with small variations in altitude. Captures at high altitude (3000+ feet) have an extremely wide-area field of view and tend to contain no significant height differences among the buildings, trees, roads, and vehicles in the scene.

The most current state-of-the-art technologies representing a scene captured by a moving camera in a 3-D model has largely been built on the work of simultaneous localization and mapping research⁶⁻⁸ as well as stereoscopy work.⁹⁻¹¹

VSFM¹²⁻¹⁴ is a cutting-edge technique that utilizes the basic components of the scale invariant feature transform^{15,16} (SIFT) keypoints to perform point matching from frame-to-frame. Extracted and matched feature locations and their correspondences allow for bundle adjustment¹³ to find 3-D point positions and camera parameters that minimize the reprojection error.¹⁷ The bundle adjustment optimization problem is constructed as a nonlinear least squares problem measured by the error between the projected point in the camera plane and its 3-D position. Similar to Refs. 18 and 19, matched points are used to compute the location and orientation of the camera associated with each frame.²⁰

The PVR (Refs. 21 and 22) creates a volumetric representation of the scene. Using an iterative scheme,^{19,21,23} a set of voxels, which are known to be surface points, is computed through a visibility map at each frame of the video. The color consistency of a voxel and visibility are used to select winners at each stage, which are added to the set of known surface points. In order to take into account the fact that voxels on the boundaries of objects may only be partially occupied, a transparency value is set to represent voxels that are partially occupied by opaque objects.²³ Within the end-result model, each voxel contains several bits of information, such as reflection, intensity, and probability of a surface point, all with respect to the viewing angle. This technique allows for the creation of complete 3-D representations (not point clouds) that provide capabilities for object recognition within the scene.²⁴⁻²⁶

Unlike VSFM and PRV, which focus on generating visually appealing models, our technique focuses on reducing the computational expenses and enhancing the density, while maintaining the usability and visual appeal of the 3-D model.

3 Scene Reconstruction Framework

We begin by examining the framework and concepts for the presented 3-D scene creation technique that reconstructs an unknown environment using only a single high-resolution moving camera. To test the framework, we assume a linear constant-speed camera path with the scene captured by a camera oriented perpendicular to the system's displacement vector. The assumptions have been temporarily placed on the reconstruction procedure to analyze its effectiveness and efficiency. To utilize the reconstruction technique on a real-time RW UAS, these assumptions must be dropped and accurate computations should replace those values.

Once a scene model is created, the distances and sizes within the model are relative. In order to determine the scale factor of the model, one must know the speed or position of the camera as well as the focal length of the camera lens for each frame. By knowing the camera specifications and speed, a disparity array that translates image pixels to

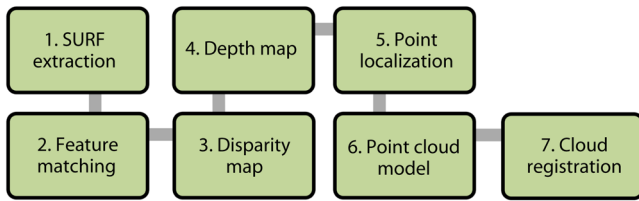


Fig. 1 Steps to create a three-dimensional (3-D) model from two-dimensional (2-D) video data captured by a single moving camera.²⁷

RW distances can be constructed. In a disparity array, various pixel distances are assigned RW depth distances and computed features' disparities are interpolated into the array to determine their corresponding depth.

The scene reconstruction framework is broken into seven sequential steps as shown in Fig. 1.²⁷ In our framework experiments, we utilize a Canon EOS 5D Mark II with a Canon EF 24 to 70 mm f/2.8L II USM Lens keeping the camera and lens setting consistent throughout the video sequence. The results in this section containing images from test videos recorded in a laboratory setting as well as RW data captured from a moving car and a UAS.

3.1 Speeded-Up Robust Features Extraction

The first algorithmic step is speeded-up robust features (SURF) extraction, wherein we locate stable feature points within each frame using the SURF algorithm. The SURF algorithm uses a Fast-Hessian detector²⁸ to identify distinct feature points within an image. By computing the determinant of the Hessian, we are able to determine the location and scale of a feature. For example, given a point $\mathbf{x} = (x, y)$ in an image I , the Hessian matrix $H(\mathbf{x}, \sigma)$ in \mathbf{x} at scale σ is defined as follows:

$$H(\mathbf{x}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{x}, \sigma) & L_{xy}(\mathbf{x}, \sigma) \\ L_{yx}(\mathbf{x}, \sigma) & L_{yy}(\mathbf{x}, \sigma) \end{bmatrix}, \quad (1)$$

where $L_{xx}(\mathbf{x}, \sigma)$ is the convolution of the Gaussian second-order derivative $(\partial^2/\partial x^2)g(\sigma)$ with the image I in point x , and similarly for $L_{xy}(\mathbf{x}, \sigma)$, $L_{yx}(\mathbf{x}, \sigma)$, and $L_{yy}(\mathbf{x}, \sigma)$.

As Gaussian filters are nonideal in any case, Bay et al.²⁸ push the approximation even further with box filters. These approximate second-order Gaussian derivatives can be evaluated very fast using integral images, independent of size.

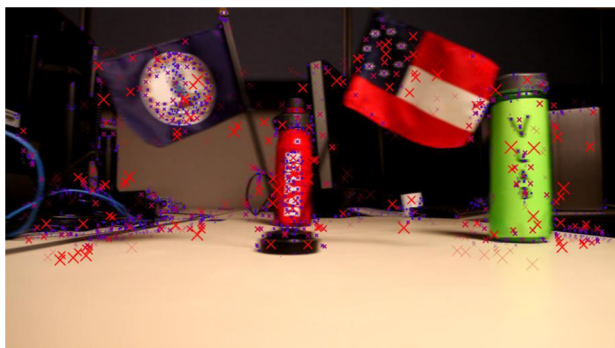


Fig. 2 Subset of calculated and trackable speeded-up robust features (SURF) points with a frame. Only points of a large scale are shown.²⁷

SURF is a local invariant interest point detector-descriptor, similar to its predecessor, the accurate but slower SIFT (Ref. 16) algorithm. We utilize the SURF algorithm due to its unique rich descriptors and relatively quick processing time allowing us to track and register points from subsequent frames. A SURF point is described by a 128-element vector, containing information regarding the feature's size, location, and orientation. Figure 2 shows a sample image with indicated SURF points from a video sequence used for 3-D reconstruction.²⁷

3.2 Feature Matching and Disparities

Once SURF points have been identified, we examine the distance each point has traveled between frames to generate a disparity value for each feature point as described in our previous work.^{27,29–34} The feature matching step compares the SURF point descriptors between adjacent frames to identify nearest matching descriptors.³⁵ The feature matcher is implemented using squared Euclidean distance to enable a bailout threshold to maximize performance. In order to determine which SURF point matches are reliable and which are spurious, we track the path of a feature across a window of frames, retaining only those points that maintain an uninterrupted path. As a feature travels from frame to frame, we set a threshold based on the previous magnitude and direction of the feature's path. The filtering process described eliminates spuriously matched features. Specifically, we establish match relationships between SURF points in adjacent frames A_i and A_{i+1} and we identify the matching points $a \in A_i$ and $b = \operatorname{argmin}_{\beta \in A_{i+1}} \|\alpha - \beta\|$, where α and β are the associated feature

descriptors of points $a \in A_i$ and $b \in A_{i+1}$, respectively. The points are separated by a maximum feature descriptor distance, $\delta > \|\alpha - \beta\|$, as well as a maximum spatial distance. Here, we use the Euclidean distance, both for the purpose of feature matching and disparity calculation; the Manhattan distance³⁶ has been shown to provide robust results, as well.

Then given a temporal window of frame size n , the set of all frame-to-frame correspondences C has size $|C| = n - 1$, where

$$C_i = \bigcup_{a \in A_i, b \in A_{i+1}} (a, b), \quad (2)$$

for all points $a \in A_i$, and where

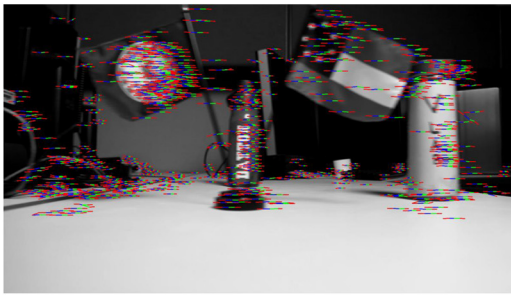
$$b = \operatorname{argmin}_{\beta \in A_{i+1}} \{\|\alpha - \beta\|\}. \quad (3)$$

In order to determine which SURF feature matches are reliable and which may be spuriously represented in the disparity map, we track the path of each feature across a window of n frames, retaining only those points offering an uninterrupted tracked path across an entire given window. That is, for all $C_i \in \{C_1, C_2, \dots, C_{n-1}\}$, we identify those point pairs $(a_i, b_i) \in C_i$ and $(a_{i+1}, b_{i+1}) \in C_{i+1}$ where $b_i = a_{i+1}$. This path tracking can be thought of as a pruning process, where points not on the uninterrupted path from A_1 to A_n are removed from the sets of matched points.

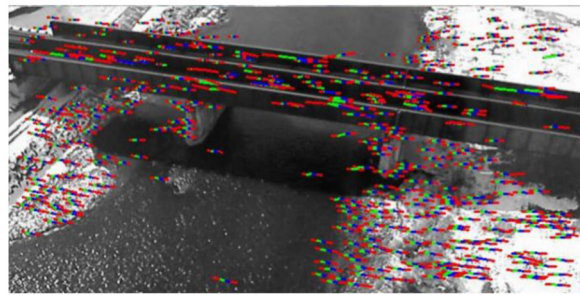
For each feature point, an array of coordinates corresponding to the feature's location (row, col) within each of the frames of a window is output and a "distance traveled" metric for the feature is computed. This distance value is referred to as a feature point's disparity. When each tracked



Fig. 3 SURF points are extracted in two consecutive video frames. Using the matching procedure described, feature points are matched and tracked from frame to frame. Input authorized for release by PRCI.



(a)



(b)

Fig. 4 A disparity map demonstrates the depth calculation concept of this framework. (a) A desk scene shows that objects closest to the camera plane, such as the flags, exhibit the largest disparities, while distant objects, such as the bottle, exhibit smaller disparities. (b) A frame from an unmanned aerial system (UAS) video demonstrates that the bridge appears closer to the camera than the surrounding banks. UAS input imagery provided by IDCAST.²⁷

feature point has been associated with a disparity, a disparity map illustrates the depths within a scene.^{37–40} In Fig. 3, we demonstrate the matching process between two frames.

As the scene shifts through the video, points are tracked from frame to frame by matching their SURF descriptors. Figure 4 illustrates disparity maps for two scenes.²⁷ In Fig. 4(a), we demonstrate a laboratory setting with the camera traveling perpendicular to the scene illustrating the disparity map of SURF points. Similarly, Fig. 4(b) shows an outdoor scene captured from a UAS. In both images, each disparity line consists of four smaller segments representing the path of each feature point from frame-to-frame within a window. Examining the disparities illustrates that feature points closer to the camera plane have large disparities, while feature points far away from the camera plane exhibit a much smaller disparity.

3.3 Depth Triangulation

In the fourth reconstruction step, we convert from two-dimensional disparities to 3-D Cartesian coordinates. We begin by converting the disparity map into a depth map by assigning appropriate depth values to each feature. The depth is displayed along the z axis in the 3-D model. The conversion is done according to the following equation:

$$\text{Depth} = \frac{(\text{Baseline}) * (\text{Focal length})}{\text{Disparity}}. \quad (4)$$

With the assumptions of constant speed and focal length, Eq. (4) describes the inverse relationship between feature disparities and their depth. When a depth, the z coordinate, for each feature point is determined, we compute the x and y coordinates using the focal length information. In point localization, we use the depth value computed for each feature point to determine the horizontal and vertical field of view at that particular depth. The x_m and y_m coordinates in the model are dependent on the x and y image coordinates and their location relative to the center of the image. The model coordinates (x_m, y_m, z_m) are determined from the following equations, where (x_i, y_i) points are the image coordinates:

$$x_m = x_i - \frac{M}{2}, \quad (5)$$

$$y_m = \frac{N}{2} - y_i, \quad (6)$$

$$z_m = \text{Depth} = f(\text{Disparity}). \quad (7)$$

Here, M represents the image width and N represents the image height.

3.4 Point-Cloud Model

At the sixth step of the reconstruction, the initial 3-D model appears as a single point cloud as shown in Fig. 5.³³ It is

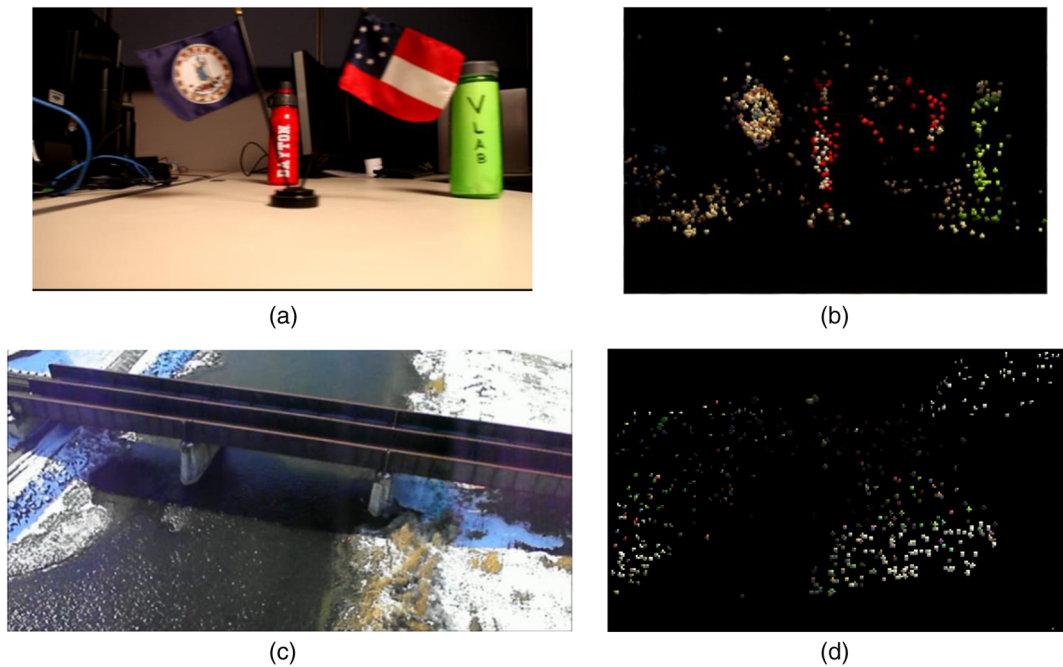


Fig. 5 (a) An original frame from a video recorded on a desk in a laboratory setting. The objects closest to the camera are the flags followed by the green water bottle and then the red bottle. (b) The single-window sparse 3-D reconstruction model of the scene presented in Fig. 5(a). (c) An original frame from an aerial UAS flight of a railroad bridge over a river. (d) A sparse 3-D reconstruction of the bridge scene. It is difficult to see any structure within the model due to the small number of points.³³

a collection of the (x, y, z) points, computed in the earlier steps, and their associated RGB values carried through from the original frame. Those values are passed along with the initial feature coordinates (x_1, y_1) and final feature coordinates (x_2, y_2) for each feature point. The colored point clouds presented in Fig. 5 show a reconstruction from a single window of frames.³³ When comparing the original frames, Figs. 5(a) and 5(c), to the generated point clouds of the scenes, Figs. 5(b) and 5(d), similarities can be observed. Each frame contains ~ 8000 detected SURF points; however, only 400 to 800 remain after the feature matching process is applied. The resulting point clouds in Figs. 5(b) and 5(d) are very sparse, but, as later results will show, the SURF points proved high accuracy with little noise.

In order to produce denser point clouds a, point registration substep is required. In this substep, point clouds generated from two different windows are meshed together into a single larger point cloud. This multiwindow point cloud contains points from every set of frames in the video sequence. The registration process compares features of various single-window point clouds to determine which features are new to the scene and which have already been registered through previous frames. The registration of features is done by comparing the feature descriptors of the SURF points. As the camera travels, most of the scene is unchanged from frame to frame, therefore, only a few new points are added as each new window is registered. As a result, a single dense multiwindow point cloud is created, which represents the 3-D model of the recorded scene. The registration process creates a global list of points from the scene.

3.5 Reconstruction Framework Evaluation

The SURF extraction implementation has been evaluated on a variety of sample videos, including UAS data and other

scenarios. The video quality varies between 720 and 1080p, which yields on the order of 8000 to 9000 SURF points per frame; however, only a subset of those points can be used for tracking the features. SURF performance is relatively slow, on the order of 4 fps on a single core and 24 fps on a dual quadcore. SURF calculations on separate frames are perfectly parallelizable. CUDASURF (Ref. 41) and other GPU-enabled implementations exist for additional performance gains and to enable a real-time reconstruction system.

Figure 6 (Ref. 27) illustrates how we have evaluated the accuracy of the reconstruction model. The accuracy is measured separately in all three directions (x , y , and z axes). Figure 6(a) shows the point cloud perpendicular to the scene similar to the camera's point of view. The yellow lines represent the vertical measurements we made within the model to be compared to the RW values.³³ The blue lines represent the horizontal measurements. Figure 6(b) shows the point cloud view from above, with the flags located closest to the camera plane, followed by the green bottle and the red bottle. The orange lines represent the depth measures. Table 1 shows the measured model units, the model units converted into inches, the RW values (truth values), and the difference between the model and the RW. From Table 1, we can conclude that the x - and y -direction computations are accurate to within an inch to the RW values, while the z direction contains the most error and uncertainty.²⁷ In order for us to evaluate the reconstruction, we use a reference object to compute the conversion factor in each direction. For example, in Table 1, we measured the true height and width of the flags to determine the conversion factors. This factor is applied to test model measurements to obtain a RW value (inches) and compute an error. Note that the measurements in the horizontal and vertical axis contain sub-inch precision. This is due to the fact that the point localization step

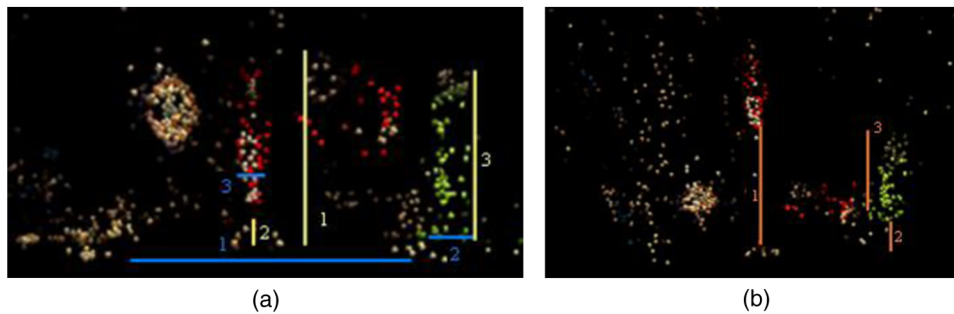


Fig. 6 (a) A perpendicular view of the point cloud with indicators on distances used in the accuracy analysis. (b) A top view of the point cloud. The measurements made for the evaluation in the x , y , and z axes are indicated with numbered lines.³³

contains less ambiguity and estimation compared to the disparity and depth computation steps. In the z axis (depth), the table indicates accuracy precision to within 1 to 3.5 in.

We observe that although SURF points are accurate and extremely distinctive, the matching process eliminates too many points. The resulting point cloud is not suitable for our applications. A more rigorous technique must focus on creating a point cloud containing more points. In the

Table 1 The table contains measured and computed values of a three-dimensional (3-D) reconstruction within a laboratory environment.

	Model (normalized units)	Model (in.)	Real world (in.)	Model-Real world (in.)
Horizontal				
Measurement Blue 1	0.1129	12.8706	13.1	0.2294
Measurement Blue 2	0.0342	3.8988	3.7	-0.1988
Measurement Blue 3	0.0324	3.6936	3.7	0.0064
Vertical				
Measurement Yellow 1	0.0842	9.5988	9.5	-0.0988
Measurement Yellow 2	0.0132	1.5048	1.5	-0.0048
Measurement Yellow 3	0.0751	8.5614	8.3	-0.2614
Depth				
Measurement Orange 1	0.1195	19.0005	22.5	3.4995
Measurement Orange 2	0.0291	4.6269	6	1.3731
Measurement Orange 3	0.0904	14.3736	16.5	2.1264

following section, we introduce additional steps into the framework architecture to create denser point clouds.

4 Dense 3-D Reconstruction

In order to create a model with more points, an additional algorithmic step needs to be included. We propose a technique in which a point cloud is generated using both global and local information. Specifically, we generate optical flow disparities using the Horn-Schunck optical flow estimation technique^{42,43} and evaluate the value of these features for disparity calculations using the SURF keypoint detection method. Figure 7 shows where the optical flow points fit into the system flow discussed in the previous section.³⁴ Optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. It can arise from the relative motion of objects and the camera. Consequently, optical flow can give important information about the spatial arrangement of the objects viewed and the rate of change of this arrangement.

4.1 Dense Point-Cloud Models

In this section, we illustrate the results of optical flow on RW images and demonstrate the enhanced effect created by adding optical flow into the reconstruction algorithm. Results of the optical flow implementation are shown in Figs. 8, 9, and 10.³⁴ In Fig. 8, we observe a frame from a street scene.³⁴ Each optical flow point is associated with a direction and magnitude, which are used as the point's disparity vector, and serve as an additional pair-wise feature for the depth triangulation. The points are also color coded

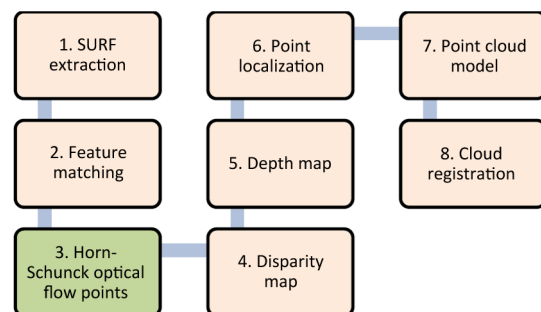


Fig. 7 Steps to create a 3-D model from 2-D video data captured by a single moving camera. The more advanced technique generates dense point-cloud models with the marked additional steps of Horn-Schunck optical flow.³⁴

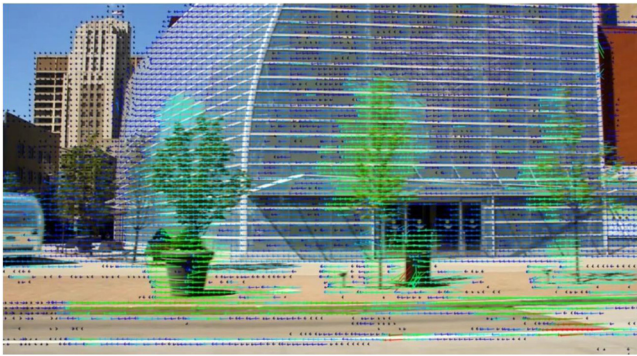


Fig. 8 A disparity map of optical flow points demonstrates the depth of feature points from the camera. An outdoor street scene shows that objects closest to the camera plane, such as the trees, exhibit the long disparities, while objects farther away, such as the buildings, exhibit shorter disparities.³⁴

based on the magnitude of the disparities. Long disparities are indicated in red, slightly shorter disparities are indicated by green, followed by blue and a single pixel disparity is indicated in black. The disparities show a strong indication of the depth of particular objects in the scene. Parts of the road closest to the camera are marked in red, the trees are in green, the building behind is in blue, and the background is in black.

These new optical flow feature points, in addition to the earlier SURF points, create a dense 3-D model. Figures 9(a) and 9(b) depict trees along the side of a road.³⁴ The image on the left illustrates the original frame, while the image on the right shows the reconstructed model. Note the density and details within the trees in the scene. Similarly, Figs. 9(c) and 9(d) demonstrate a bank building, which produces a model dense enough to be able to read the name on the building.³⁴

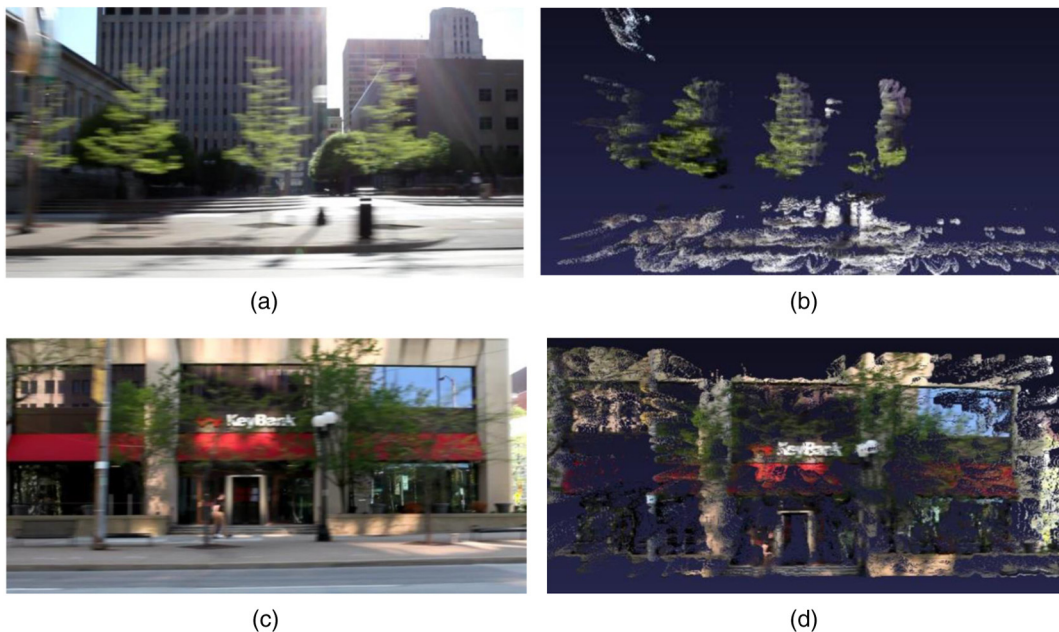


Fig. 9 The point clouds generated utilizing the Horn-Schunck optical flow method. (a) and (c) show the original input frame, while (b) and (d) are reconstruction shown at a perpendicular angle as the camera travels along a city street.³⁴

4.1.1 Reconstruction from aerial imagery

Challenging videos captured from a UAS tend to contain significant jitter, instability, and rapid orientation changes. Figures 10(a) and 10(b) show an aerial scene.³⁴ This frame comes from a light-weight UAS recording video of a railroad overpass. In Fig. 10(a), we see the original frame from the video, while in Fig. 14(d), we observe the 3-D reconstruction model viewed from the same angle as the original image. Note the visibility of the tracks and depressions that occur in the scene. Figure 10(c) shows the same point-cloud model rotated 90 deg. From the profile view, we can see that the green grassy part of the scene represents a hill and the incline is noticeable from the model. As the point cloud rotates, the elevation for the hill becomes apparent as well as the variation in size of columns supporting the railroad. Interestingly, the subtle incline is not apparent in the original image, but significantly stands out in the 3-D model.

4.1.2 Interior reconstruction models

In another test scenario, a set of indoor video sequences has been captured from a mobile robotic platform, a UGV, and used to reconstruct an interior scene. An illustration of the scene captured is shown in Figs. 11(a)–11(d). In the video frames, we illustrate the front lobby of Kettering Laboratories at the University of Dayton. The scene was captured by first moving in one direction with the camera positioned nearly perpendicularly to the direction of motion. In the scene, the major areas of interest are the two elevator doors, which are slightly indented into the surrounding walls, and the trash/recycling bins. The objects within the scene are a variety of depths. We were interested in observing if the small depth variations, such as the elevators being further away from the camera plane than the surrounding walls, would be present in the 3-D model. In the next figure, Figs. 11(e)–11(h), we describe the optical flow patterns

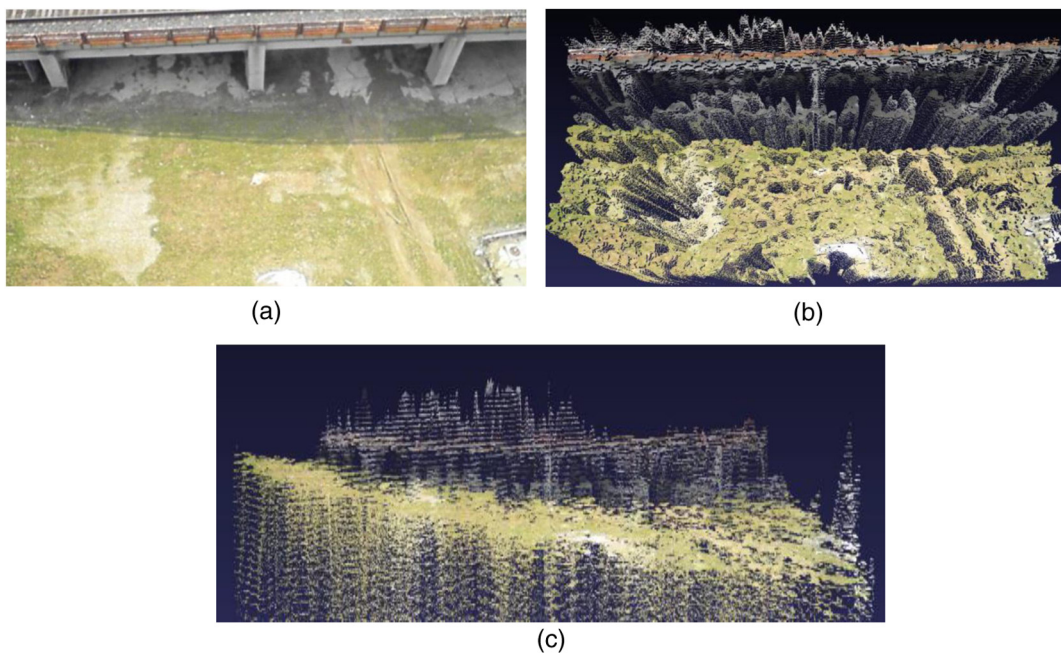


Fig. 10 (a) Original video frame, (b) reconstructed model viewed from the same angle, and (c) when the model is rotated the incline of the hill becomes visible with the railroad horizontal in the background.³⁴

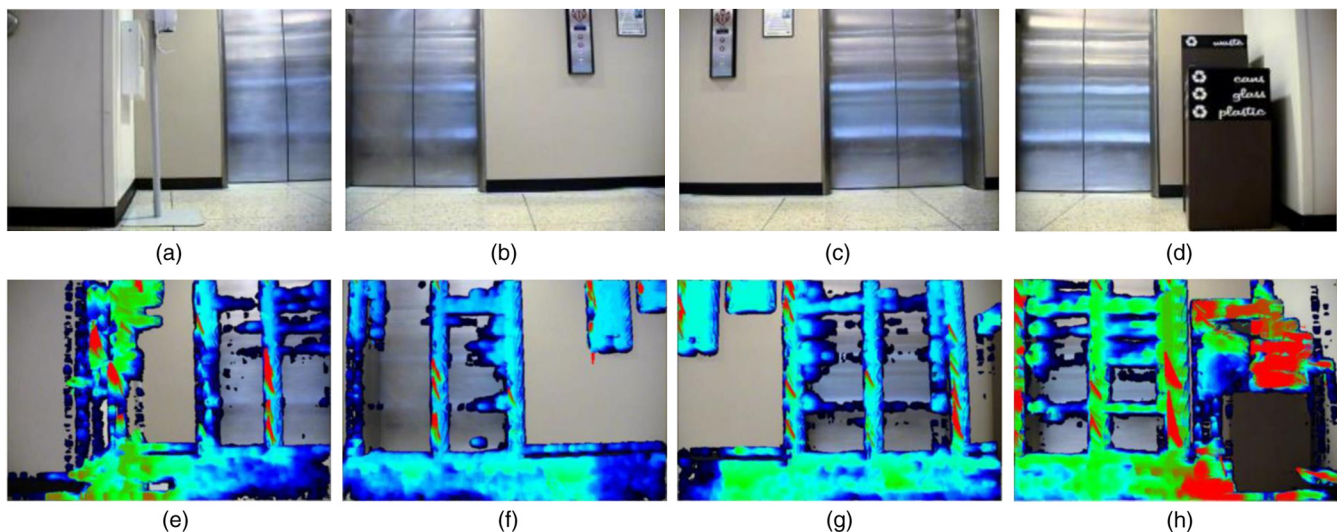


Fig. 11 Sample frames in sequential order (a) to (d) from a video capture by a mobile platform. The images in sequential order (e) to (h) illustrate the disparity map of the optical flow points. These disparity maps are color coded, where red (long arrows) represents points closest to the camera followed by green (medium-length arrows) and blue (short arrows), respectively.

associated with each frame. Once again, the color coding indicates the lengths of the velocity vectors or disparity of the point. Red marks the longest disparities and corresponds to objects closest to the camera plane, followed by green with slightly shorter disparities, and, finally, by blue with the shortest disparities in the scene.

The 3-D reconstruction model of an interior hallway is presented in Fig. 12. We show the success of the reconstruction framework in determining the depth of feature points within the scene. Figure 12(a) illustrates the compilation of the original frames to create a sense for the entire RW

scene. Figure 12(b) demonstrates a point cloud composed of 435,447 points of the elevator scene. When observing the reconstruction model, several things stand out. First, there are large vacant spaces where a wall or elevator door should be located. This is due to the textureless nature of some objects in the scene. For example, the walls around the elevator doors contain no distinct texture. Therefore, features from parts of the wall are unable to be correctly matched in subsequent frames. Similarly, parts of the elevator door contain no identifiable texture, therefore, the disparity values of those features are inaccurately small. Extremely small disparity

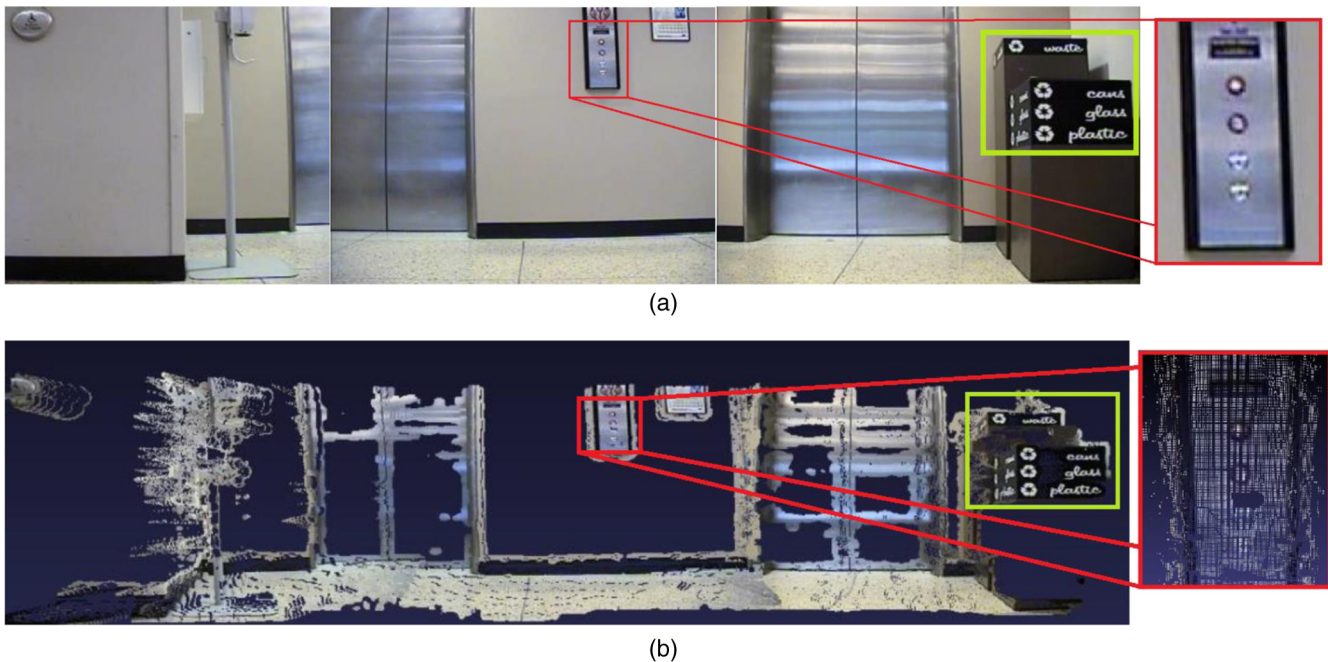


Fig. 12 (a) Several frames from the input video concatenated together to illustrate the real-world scene. (b) The 3-D point-cloud model of the scene created from the reconstruction framework described.

values would cause the feature points to appear extremely far away. Second, the optical flow points have provided a much denser point-cloud model. The density allows the user to observe and recognize small scene details, such as the elevator buttons highlighted by the red box in Fig. 12. To illustrate the details a dense point-cloud model provides, we compare the original image to the reconstruction model. Our selection is emphasized with green boxes in Fig. 12. We have selected a part of the scene that contains writing to show the density of the cloud. The optical flow density allows for signs within the model, such as the cans, glass, and plastics on the recycling bin, to be legible.

To illustrate the accurate depth computation within the model, we focus on the small detailed depth differences in the scene. First, we observe the difference between the elevator doors and the surrounding walls. Notice that in the RW scene, the elevator doors are indented several inches into the wall. In Fig. 13(a), we illustrate a view from above the model. The figure focuses on the corners between the elevator doors and the surrounding wall, marked by red boxes, where it can be noted that the two are correctly placed in separate depth layers. In Fig. 13(b), we show two additional examples of the wall corner being dense enough to block the view of the elevator doors when the model is rotated.

4.1.3 Reconstruction from ground imagery

We conclude the result demonstrations with what has become the signature model of the optical flow 3-D reconstruction algorithm. This model is constructed using a video recorded from a car driving on Main Street in downtown Dayton, Ohio. The captured scene contains various details, including buildings, trees, benches, light poles, a parking lot, etc. The model displayed in Fig. 14(b) consists of over two million feature points. These points are dense enough to make out distinct objects within the scene.

Figures 14(c) and 14(d) demonstrate an angled view of the street and an overhead view of the street, respectively.

We also illustrate the added density and depth variation. For example, in Fig. 15(a), a zoomed-out view of the building allows the viewer to clearly distinguish the Performance Place sign as well as the plant pots, light pole, and fire hydrant closer to the camera. To illustrate depths more clearly, we focus on the white car parked in the parking lot behind the trees. In Figs. 15(b) and 15(c), the car is marked by a red box. From a perpendicular view, it is difficult to distinguish the depth at which the car is located relative to the trees in front. However, when we rotate the point cloud, notice how the car disappears behind the trees. Similar to a human's daily experience of obstruction of view when closer objects obstruct one from seeing the scene behind the object, this model can be rotated and examined in any angle.

4.2 Metrics, Analysis, and Evaluation

The reconstruction algorithm has been evaluated on a variety of sample videos, including data captured by a UAS and a UGV in indoor and outdoor environments. We plan to compute the accuracy of the reconstruction model by measuring the RW values in all three directions (x , y , and z axes). We begin by evaluating the horizontal direction of the street point cloud. We made 10 RW measurements and compared them with the same 10 measurements within the model. As expected, due to the eye appeal model angle, the horizontal direction within the model exhibits almost the exact values as measured in the RW. Figure 16 marks the 10 measurements made within the point cloud and charts the differences between the model and RW values.³⁰ Similarly, Fig. 17 illustrates the measurement conducted in the vertical direction.³⁰

The third accuracy metric is evaluated in the depth direction (z axis). As shown and charted in Fig. 18, 10 RW measures were taken and compared with the same measurements

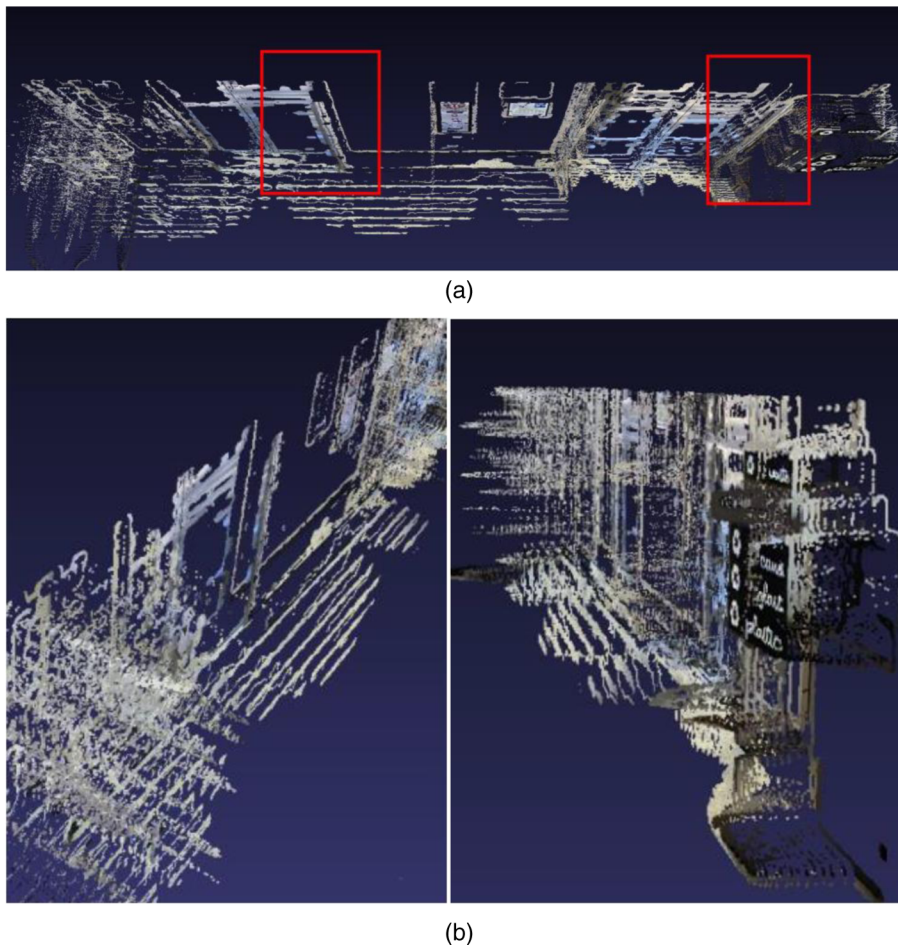


Fig. 13 (a) A view from the top of the 3-D point-cloud model. Note that different objects are correctly placed in different depth layers. (b) Several views of the 3-D model illustrating the density of the point cloud.

within the point cloud.³⁰ Unlike the horizontal and vertical directions, the depth exhibits slightly higher error measurements. The conversion factor and average error in feet for each direction are shown in Table 2. The table indicates that converting the relative model values into RW coordinates yields an average error of 1 foot in the horizontal and vertical direction and 5 feet in depth estimation.

4.2.1 Layer effect

When analyzing and evaluating the performance of the reconstruction, we consider the accuracy with relation to RW values. As indicated in Table 2, the horizontal and vertical measurements experience high accuracy (< 1.5 feet of average error).³⁴ However, the depth experiences up to 4.7 feet of error. By taking a closer look at the reason for this increase, we determine that the depth of the point-cloud model is directly related to the variance of the disparities. (Note that disparities are directly related to the image resolution and baseline between frames.) Given the resolution of the original image, the algorithm cannot distinguish between, for example, 2 and 4 feet of depth because both points get assigned the same z coordinate. As a result, when multiple points are assigned the same depth coordinate, the point cloud experiences a discrete number of depth layers. We define the layer effect as a state in which the point-cloud

model experiences discrete depth layers. This is clearly visible in figures that display an overhead view of the model, as demonstrated in Fig. 19.³⁴

By adding optical flow features in the algorithmic framework, we were able to create a dense point-cloud model. As the results indicated, the new point clouds allow the user to distinguish features of the scene by providing enough density to identify buildings, signs, and trees. With the introduction of a denser point cloud, we also introduce the layer effect, in which numerous points contain the same depth coordinate. In the next section, we propose a solution to this issue by increasing the depth resolution. We do this by incorporating a resolution enhancement technique that is applied to the input imagery.

5 Depth Resolution Enhancement

In this section, we introduce the novel concept of depth resolution enhancement.^{44,45} In order to reduce the layer effect caused by discrete and limited disparities described at the end of Sec. 4, we apply a super-resolution technique to the original frame.³¹ By increasing the resolution of the input frames, we create more layers, thus eliminating the discreteness. Due to the increased resolution, more feature points are obtained to create a more dense point cloud as well as a larger number of layers. In general, a super-resolution

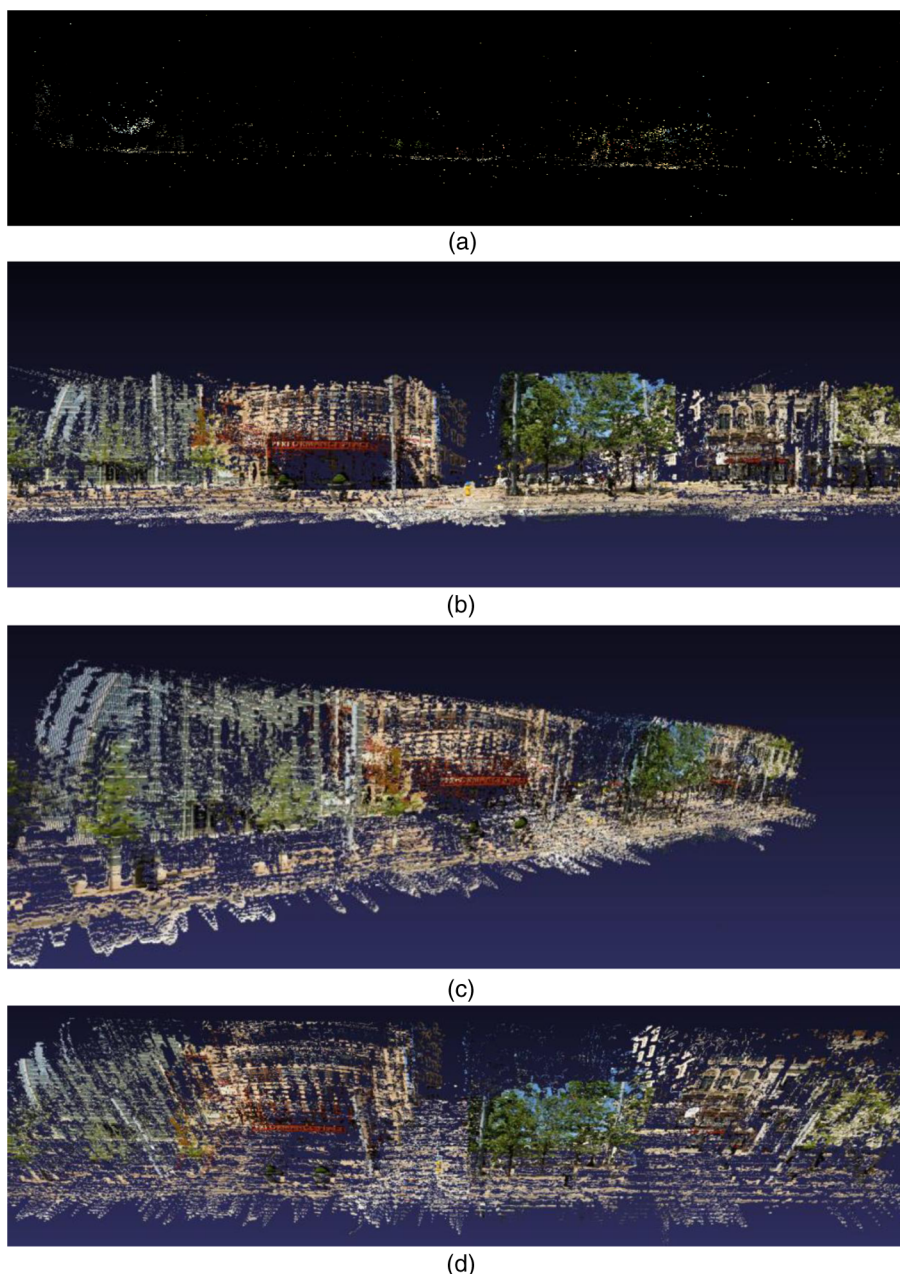


Fig. 14 Different views of a point-cloud model generated from a street scene depicting dense point clusters as objects at various depths. (a) A SURF points-only model without the optical flow points, (b) frontal view of the street in the SURF+optical flow model, (c) profile street view, and (d) top overhead view of the street.

technique^{46–48} is the process of obtaining a high-resolution image from an image or a sequence of low-resolution images. Traditionally, this process takes place by using multiple cameras mounted closely together and oriented in the same direction. Thus, a higher-resolution image can be obtained by combining pixel information from each of the camera's imagery. Our method uses a single frame to improve the quality of the image by interpolating the image. Unlike the standard linear interpolation⁴⁹ and bicubic interpolation,⁵⁰ this super-resolution technique uses the image directional variance to determine the inserted pixel values. Although computationally expensive, i.e., for C/C++ implementation, ~600 s on a dual core processor on a HP Z600 desktop station with

8 GB RAM are required for a 10 Mexapixel image to become a 160 Megapixel image, the addition of super resolution will create more disparity resolution and, thus, more depth resolution. In Fig. 20, we illustrate the additional preprocessing step in the existing architecture. We determined that the ideal fit of super resolution within our algorithm would be in the very first step prior to any feature extraction.

5.1 Analysis of Super Resolution

The original resolution image is enhanced by a resolution factor r . For each 9×9 neighborhood patch in the image, we calculate the real and imaginary components of the

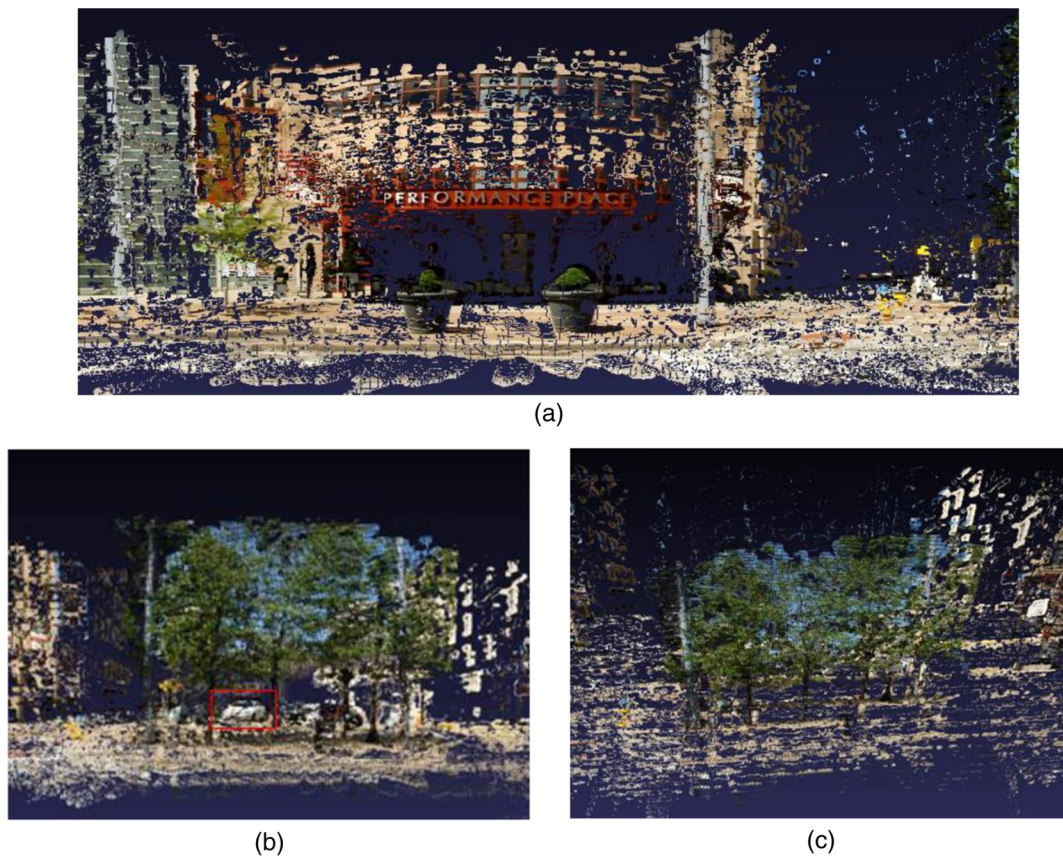


Fig. 15 Details of the point-cloud model illustrating the density and depth layers. (a) Legible Performance Place sign, (b) vehicle visible underneath trees, and (c) when the model is rotated, the trees cover the vehicle.

Fourier phase angle and normalize them. Next, the algorithm calculates four variances, 4×1 vectors, taking all 9 pixels along the horizontal, vertical, and two diagonal directions about the center pixel for both real and imaginary components. For the same patch, the algorithm estimates four variances, also 4×1 vectors, taking only the r 'th pixels along the same respective directions for real and imaginary components. Next, the covariance of the two vectors is calculated. This gives a covariance matrix at the center pixel. By adding the two covariance matrices found, the kernel

function is able to be computed. The algorithm estimates unknown pixels using kernel regression. The result is a high-resolution image of order r . We analyze the effect of super resolution on the effectiveness of various components in our reconstruction technique.

As described in the referenced algorithm,^{44,45} we use feature based covariance learning for adaptive kernel regression. As a result, a low-resolution input image becomes a higher-resolution image through nonlinear interpolation of the original pixel intensities.

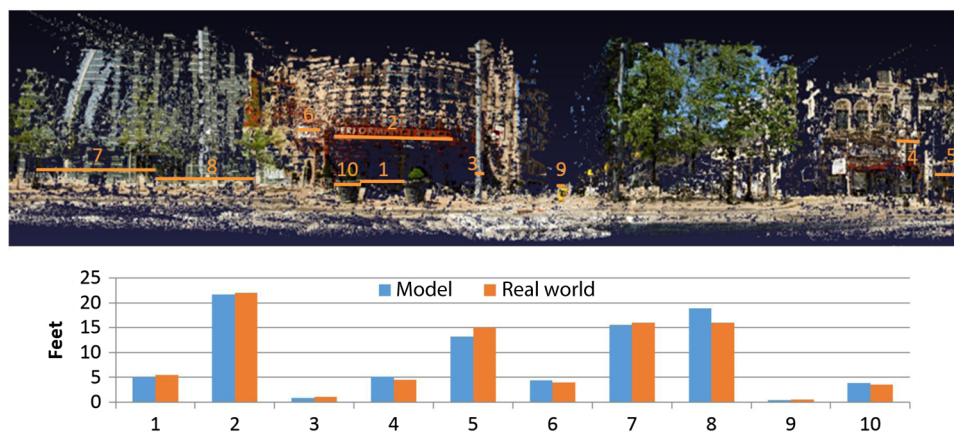


Fig. 16 Illustration of the point cloud perpendicular to the plane of the camera.³⁰

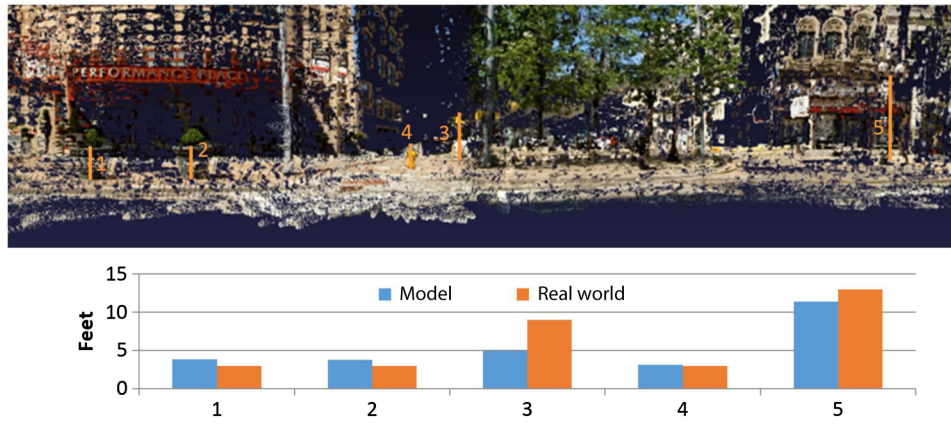


Fig. 17 The orange lines represent the vertical measurements made within the model.³⁰

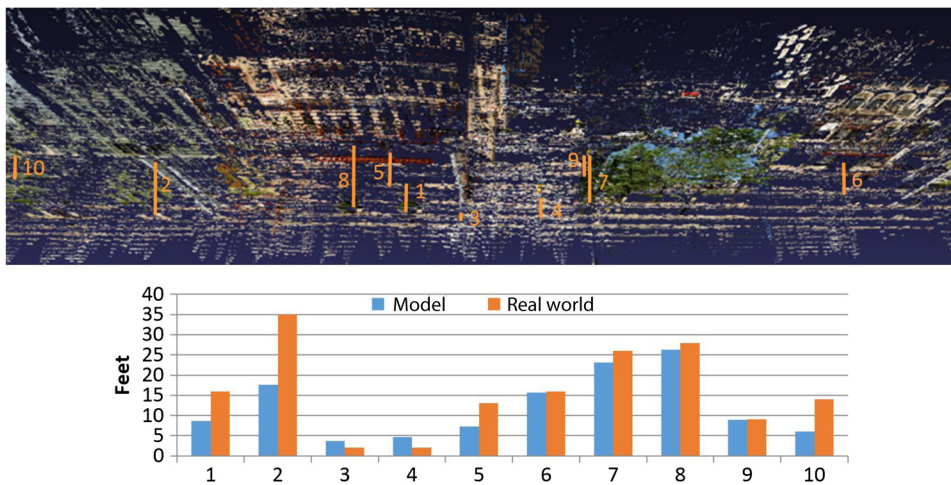


Fig. 18 We illustrate the point-cloud view from above. The orange lines represent the depth measurements that were compared to the real-world values.³⁰

5.1.1 Super-resolution evaluation

We evaluate the effects of super resolution by observing the effects on feature extraction,³⁵ such as Harris Corners,⁵¹ SIFT,^{15,16} and SURF points.²⁸ We illustrate the improvement in the number and quality of features as a result of super resolution.

Significant work was put forth into evaluating single-image super resolution in an academic sense; that is, evaluating it in the same manner super-resolution algorithms are often investigated and compared in published literature. A common way to compare super-resolution algorithms in the literature is to downsample the original image, perform super resolution, and compare to the original image (and other

methods). In our case, the comparison included mean square error and number of Harris Corners generated. This was done for several scenes, including Blue Devil,⁵² Columbus Large Image Format (CLIF),⁵³ and Full Motion Video datasets. Figure 21(a) shows the Harris Corners generated on the original wide-area image, and Fig. 21(b) shows the Harris Corners generated from an $r = 2$ super-resolution image. In all cases, the super-resolution images extracted more Harris Corners and features than the original input imagery.

The results are quantified for single-image super resolution and its effects on feature extraction in Table 3. The table compares the number of extracted features for the original image, output of the proposed super-resolution technique, output of a standard bicubic interpolation, and a linear interpolation output. We utilized three sets of data to confirm the final effect of the proposed technique. Note that in all cases, the proposed super-resolution technique produced the greatest number of extracted features. Also, interestingly, bicubic interpolation and linear interpolation produced fewer points than the original image due to the blurring effects of those algorithms. In order to determine the proper size increase to use for ($r = 2$ or $r = 4$ of original image size) super-resolution imagery, we performed several experiments to test the effectiveness of $r = 2$ or $r = 4$ on the reconstruction.

Table 2 The relative accuracy of the 3-D point-cloud models.³⁴

	Conversion	Average error
Horizontal	1 model unit = 0.0206 feet	0.7612 feet
Vertical	1 model unit = 0.0204 feet	1.4750 feet
Depth	1 model unit = 0.0206 feet	4.7809 feet

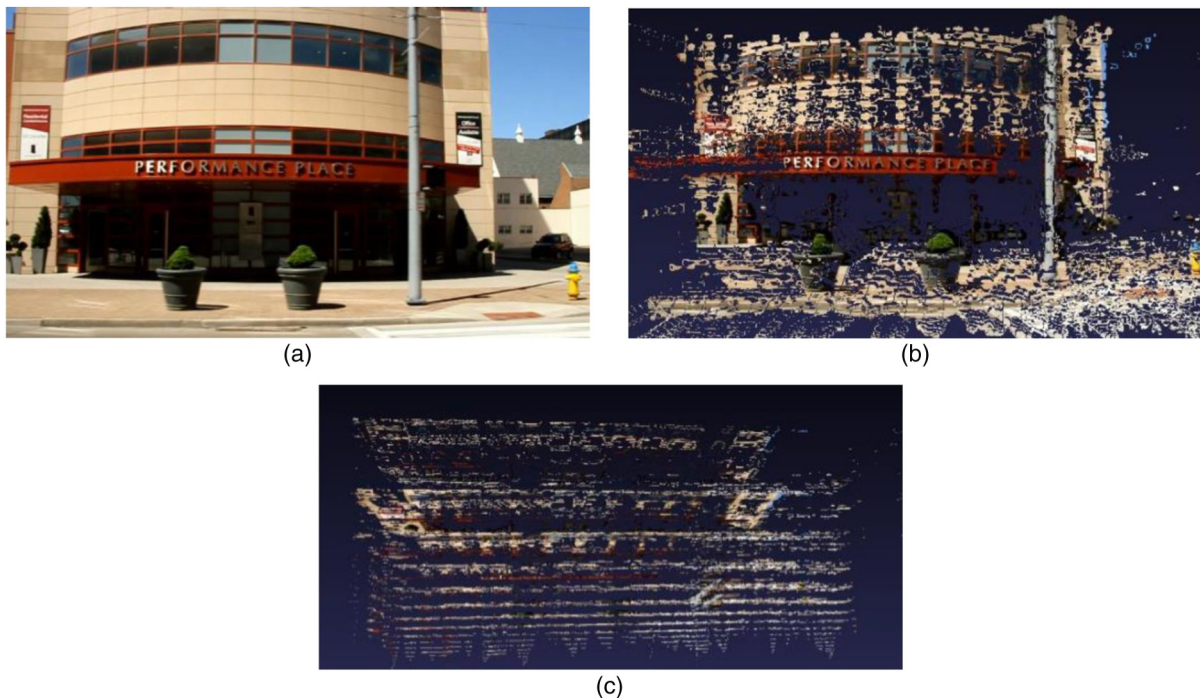


Fig. 19 (a) An original frame from a video recorded of North Main Street in downtown Dayton, Ohio. The scene contains numerous objects of different color and texture at varying depths. (b) This model is computed from two consecutive frames of video. (c) A view of the same 3-D reconstruction from the top. Note that due to the discrete nature of feature disparities, points are dispersed into layers.³⁴

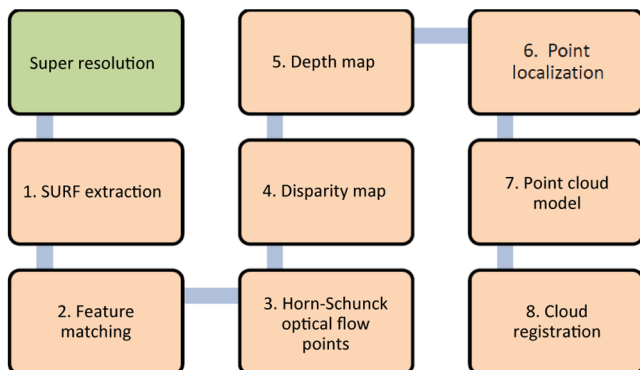


Fig. 20 The algorithmic architecture with the addition of super resolution designed to enhance the depth resolution of the point-cloud models.³¹

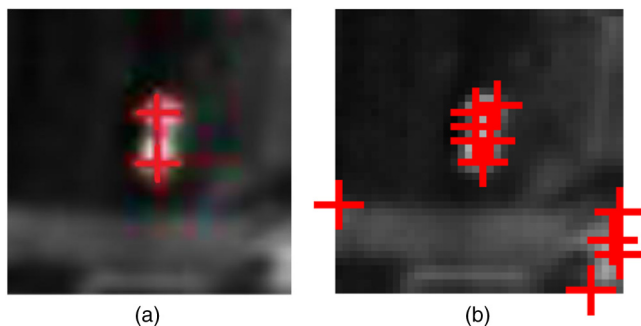


Fig. 21 A vehicle contains more feature points in the higher-resolution image: (a) original resolution input and (b) super-resolution input.

We continue the analysis by observing the effect of super resolution on an original input frame. In Fig. 22, we have cropped and scaled a small portion on the original scene to highlight the effects of super resolution.³¹ Figure 22(a) illustrates the original image. Note the effects of super-resolution edges between light and dark areas of the image. In Fig. 22(b), we have increased the resolution by a scale of $r = 2$. That is, by increasing the height and width of the image by two from 1220×720 to 2440×1440 pixels, we have created an image of four times the number of pixels as the original. In Fig. 22(c), we observe the effects of super resolution as we increase the resolution by 16 times the original. That is, the height and width of the original image are increased by four from 1220×720 to a 4880×2880 input frame.

As observed in Fig. 22, the number of pixels in the input increases while maintaining the spatial and pixel intensity relationships of the original input. As a result, the disparity resolution of tracked features also increases. Previously, the

Table 3 Number of Harris Corners extracted in three datasets after applying various interpolation techniques. The super-resolution algorithm described provides the highest number of extracted points.

Corner extraction (X2)	Original	Super resolution	Bicubic interpolation	Linear interpolation
Calibration strips	12	23	8	5
Full motion	18	48	4	2
Blue Devil	14	26	14	4

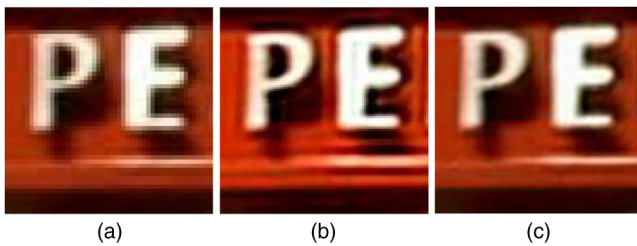


Fig. 22 The effects of super resolution are illustrated. (a) A cropped patch from original input frame that is 1220×720 . (b) Resolution is increased by a factor of four after super resolution is applied to the original input. The input image has become 2440×1440 . (c) Resolution is increased by a factor of four after super resolution is applied to the original input. The input image has become 4880×2880 .³¹

limited resolution created discrete disparities that produced the layer effect. When multiple feature points traveled the same discrete distance from frame to frame, they were associated with the same disparity value and eventually depth layer. With the super-resolution technique, the feature matching procedure was able to more precisely match the new location of the features. Therefore, while previously a feature point was matched with one particular location, after applying super resolution, that final location was represented by 16 pixels instead of one. This more precise point matching procedure created more unique disparities and, therefore, more layers within the point-cloud model.

In Fig. 23, observe the effects of super resolution onto a single-window point cloud. In this scene, the building labeled Performance Place has a rounded convex frontal wall and overhang. In Fig. 23(a), we have reconstructed the scene using the original resolution. Observe that the convex wall and overhang are depicted as a flat surface. This is caused by all parts of the wall experiencing very similar disparities and all parts of the overhang also experiencing very similar disparities. After applying super resolution to the input image, we obtain the point cloud depicted in Fig. 23(b). Note that the red overhang is now experiencing a convex curvature as well as the frontal wall of the theater building.

In the following section, we will illustrate how the results of super resolution apply to the reconstruction procedure and

Table 4 Comparison of point-cloud models created from the original input models to models created from the super-resolution inputs.

Scene	Original input		Super-resolution input	
	Number of points	Number of layers	Numbers of points	Number of layers
Scene 1	230,402	221	1,512,776	898
Scene 2	1,152,012	248	2,632,834	976

analyze the effects with relation to the model's density and the elimination of the layer effect.

5.2 Enhanced Dense Reconstruction

The results are evaluated by comparing the point-cloud model created from the original frame versus the model created from the super-resolution frames. In this section, we observe that the point cloud has more discrete disparities and the number of layers in the model increases when super resolution is applied to the input. In Table 4, we analyze two scenes.³¹ Scene 1 corresponds to the single-window point cloud created from the scene shown in Fig. 24. This scene contains a theater building, a light pole, decorative bushes, fire hydrant, etc. Scene 2 represents the entire city block illustrated in Fig. 14(b). This scene contains several buildings, trees, parked cars, etc. As shown in Table 4, for scene 1, the super-resolution technique increased the number of points within the model by a scale of 6.56. It also increased the number of layers by a scale of 4.06. In the larger scene 2, the increase in points was more subtle. As shown, the number of points within the model increased by a scale of 2.28 and the number of layers increased by a scale of 3.93. The increase in the number of points is affected by the cloud registration procedure. As point clouds continue to be rendered together, fewer new layers are introduced with each new point-cloud registration.

We examine the difference between the original point-cloud model and the new point-cloud model created from the super-resolution input images. The models are of an entire city block. In Figs. 14(b) and 14(d), we observe the

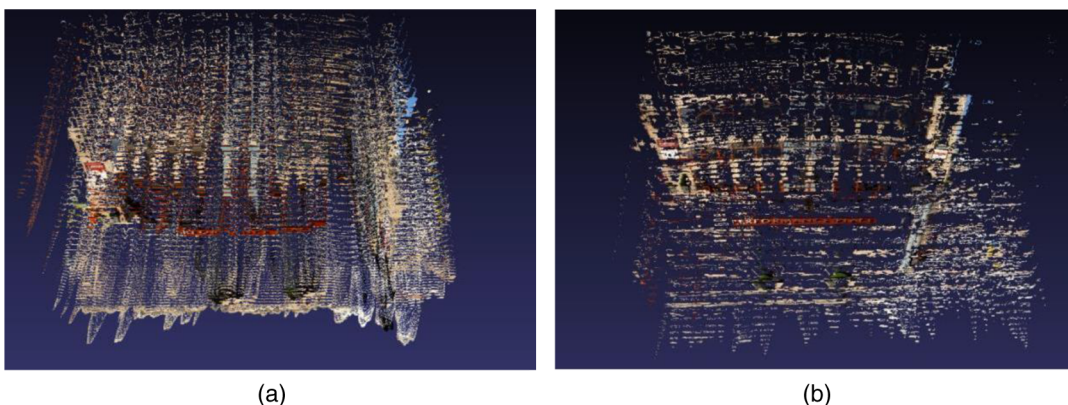


Fig. 23 The effects of super resolution on the point-cloud models. (a) 3-D point-cloud model from the original input image and (b) 3-D point-cloud model from the super resolution input image. The resolution has been increased 16 times that of the original creating more points in the point cloud and more unique layers.^{31,33}

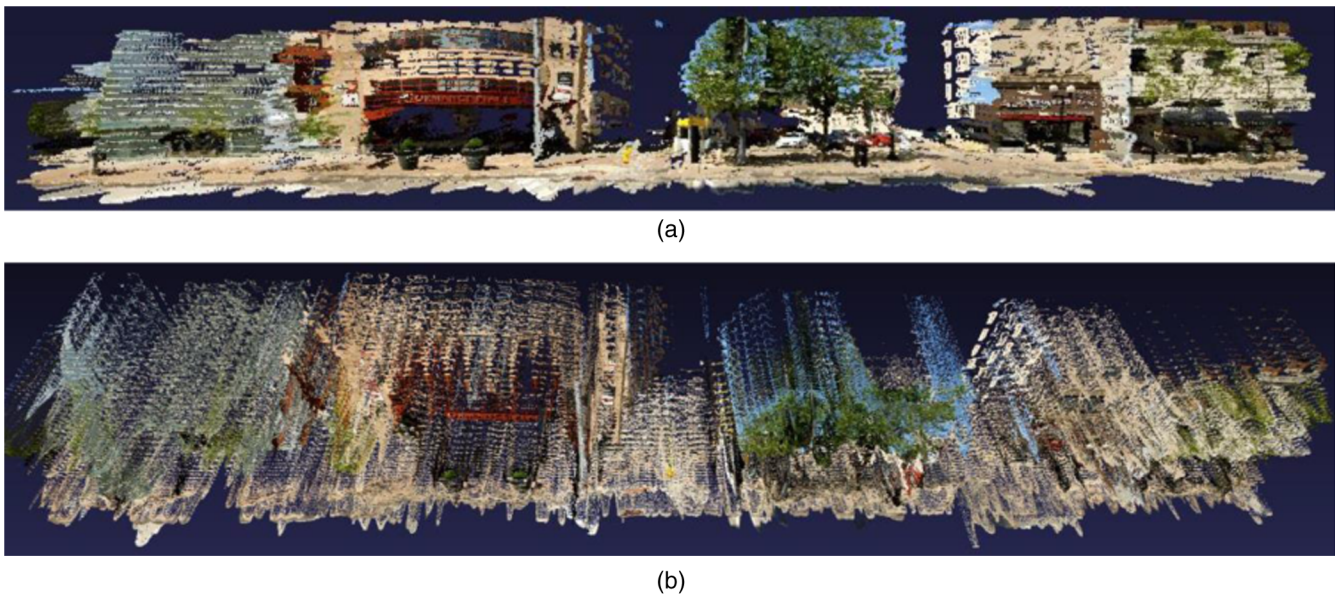


Fig. 24 (a) Frontal view of the 3-D reconstruction model created from the super resolution input frames and (b) top view of the 3-D reconstruction model created from the super-resolution input frames. Note the difference in the number of layers and density of the point cloud.³¹

point-cloud model created from the original size frames. In Figs. 24(a) and 24(b), we observe the point-cloud model created from the super-resolution input images.

The super-resolution algorithm presented produces high-resolution input frames that feed into a 3-D reconstruction algorithm. Super resolution has significantly improved the density of the point-cloud models as well as provided elimination of the layer effect caused by limited discrete disparity values. Although the super-resolution algorithm has enhanced the important features of the model, it has also magnified the noise of the model. This can be clearly observed in Fig. 24(b).³¹ The overhead view illustrates the increase in depth layers, an improvement over its predecessor in Fig. 14(d).³¹ However, as mentioned with regard to noise, notice the elongated halos around objects. This noise is present in the previous versions of the point cloud, but is much less noticeable. In the previous version, Figs. 14(b) and 14(d), little noise speckles are present throughout the scene. With the addition of super resolution, these noise speckles have also been enhanced and appear much denser and over several layers. We discuss the development of a noise suppression technique in the next section.

6 Noise Suppression of Point Cloud

In this section, we present several noise suppression techniques used to remove or reposition incorrectly placed points within the 3-D point-cloud model.³² Once the point-cloud model is produced, we would like a way to eliminate points based on their 3-D neighborhood surroundings. We referred to this process as postprocessing noise suppression. In 3-D space, points belonging to the same object will experience similarities with regard to its color and texture information. Using this texture and location information, we can filter out outliers that appear as noise. Figure 25 illustrates how the additional algorithmic steps fit into the complete DPR architecture.

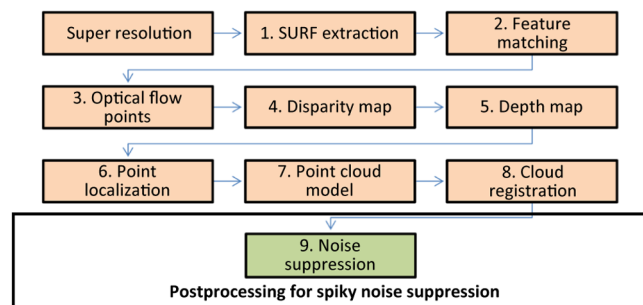


Fig. 25 The algorithmic flow diagram of dense point-cloud representation (DPR) technique. Additional noise removal steps presented in this section are highlighted in green.

6.1 Postprocessing for Spiky Noise Suppression

In this section, we present a noise suppression technique that eliminates unwanted and mismatched points from the point-cloud model.³² We begin by identifying the reason and source of noisy points. Noise is created when a feature point is mismatched from frame to frame. When a mismatch happens, the disparity value of the feature point is an outlier with respect to its neighbors. This causes the point to appear isolated in the point cloud. Due to the effects of super resolution, the number of unwanted and mismatched points has increased due to the direct relationship with the input image resolution. A mismatch occurs when the matching algorithm determines that the best match for a particular feature is not the same feature in the subsequent frame. Mismatches have a high frequency of occurrence in regions with little texture. These regions have feature points with very similar descriptors. Points from regions of similar texture can easily be mismatched within the region as well as with points in other similar regions. In order to maintain only regions of uniqueness, we perform Canny edge detection^{54,55} on the image. In this way, we maintain the important and unique aspects of

the scene while eliminating the regions that cause mismatches. We can fill in these regions once the depth values of the edge have been established. Using edge information alone, we are still able to correctly interpret the scene and distinguish depths.

We present a technique that suppressed the spikes and scattered points from textureless regions. The noise associated small mismatched regions are called spiky noise. Due to the small size of the region that is being mismatched, the points within the region follow a curvature that peaks in the center of the region. Notice these spikes in the point-cloud model presented in Fig. 24. The technique presented below eliminates those spikes from the model.

We illustrate the removal process in Fig. 26. We have created a filter to cluster the point-cloud model into facades. By clustering the model into distinct N facades, we are able to perform image noise removal techniques. In Fig. 26(a), we illustrate one of those layers.³² It is composed of thousands of points clustered together at a particular depth within the model. The resulting image is mostly black; however, the (x, y) coordinates for points within the specified region are displayed. We show the extent of these points in Fig. 26(b) by applying dilation.³² Notice that small speckles are floating between larger objects in the scene. By performing the opening morphological operation, we are able to eliminate those noise points. The opening operation, shown in Fig. 26(c),

is composed of first eroding the image with a structural element.³² Then, to regain the losses of the erosion, we apply dilation with the same size structuring element. Noise smaller than half the structuring element is eliminated through this technique. Figure 26(d) shows the final result.³² Our final step uses the original depth information to return the model to 3-D space. Only unfiltered points are plotted back in the model. The noise suppressed resultant point-cloud models are presented in the following section.

6.2 Filtered Point-Cloud Results

We present the final point-cloud model results. These models represent the resulting work of the DPR. The evaluation is best conducted by observing the effects on the actual point cloud. In Figs. 24(a) and 24(b), we illustrate the earlier model of the DPR.^{32,33} We analyze the 3-D point cloud models to evaluate the effects of the noise suppression techniques. The same model is processed through the noise suppression and is shown in Figs. 27(a) and 27(b). It is noticeably less noisy in areas prone to spiky noise. This technique is able to handle noise scatter over many depth layers. The attractive features generated from the reconstruction framework and super-resolution enhancement model are still present in this noise suppression model. Objects in the scene are correctly dispersed across multiple depth layers. For instance, the road and sidewalk in the scene are flat in comparison to the trees

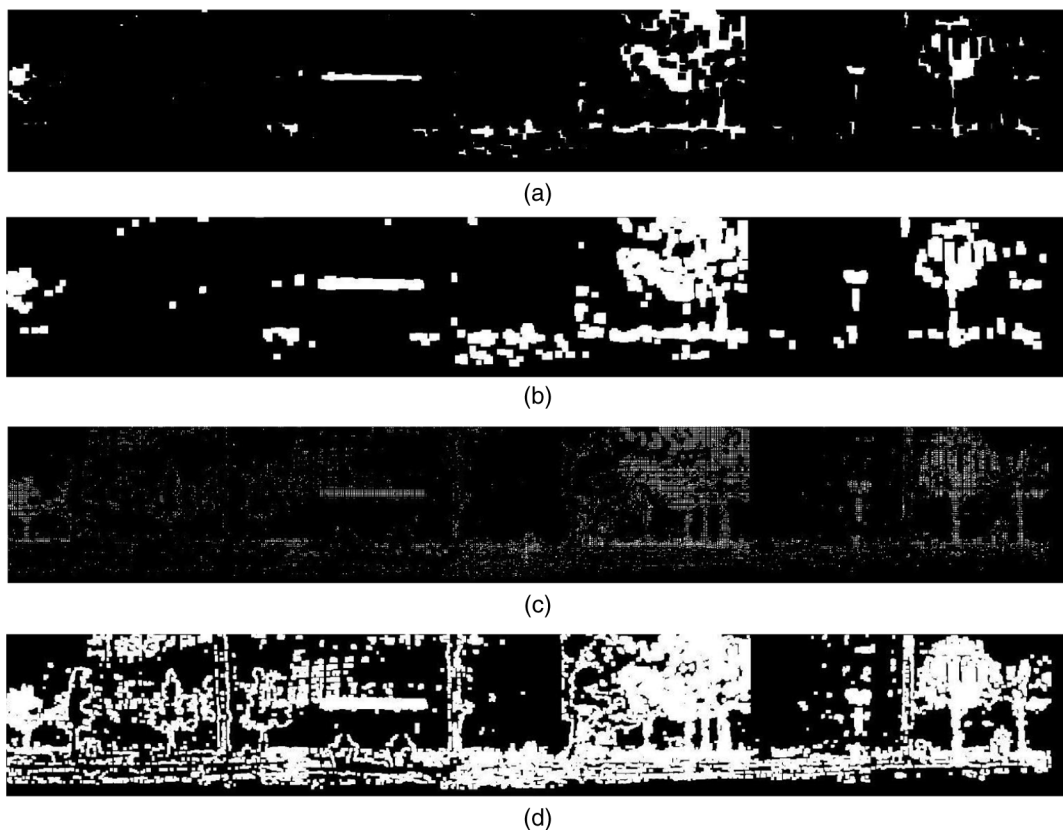


Fig. 26 (a) By clustering multiple depth layers into one, we are able to obtain a 2-D image in which spiky noise appears as small speckles. (b) We illustrate the clustering of multiple layers into one, which contains true points as well as many noise points. (c) The morphological opening operation is erosion, in which a structuring element removes all the noise and edge of larger objects. (d) The second stage of the morphological opening operation is dilation.³²

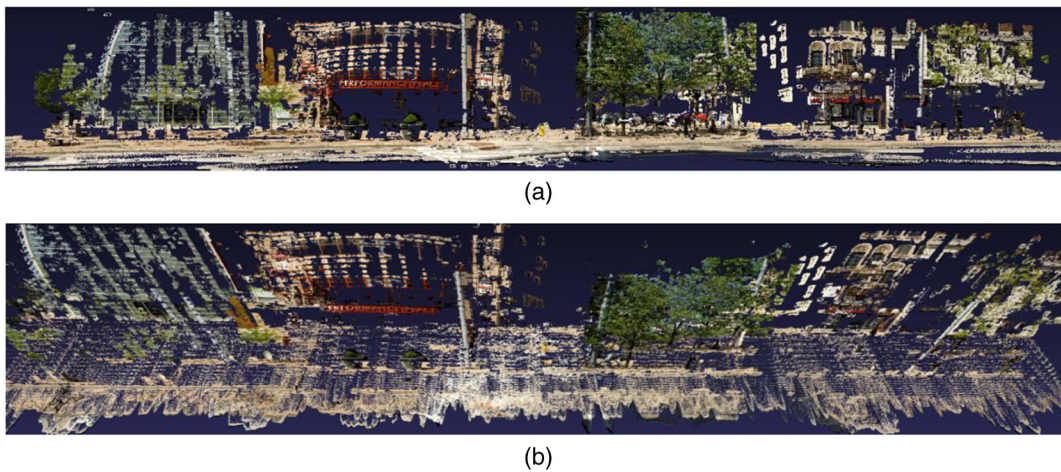


Fig. 27 (a) The point-cloud model after the noise suppression processing. All the major features of the scene are retained but little noise speckles are eliminated using the technique. (b) A view of the new clean point-cloud model. This model features a variety of depth layers and dense reconstructed objects from the input scene.^{32,33}

and buildings. The density of the point cloud remains over 2 million points.

In order to appreciate the results of DPR, in Sec. 7, we compare the reconstruction model to other representation techniques. In the following section, we describe the experimental design and evaluate our proposed technology versus the state-of-the-art techniques. A detailed discussion will summarize the advantages of the proposed method and potential use for this technology.

7 Performance Evaluation

We revert back to some of the techniques discussed in Sec. 2 and compare the results of our DPR algorithm with two rivaling techniques. All these 3-D modeling techniques have the promise of aiding systems in the areas of change detection, contextual information such as elevation, roads, georegistration, detection, and elimination of shadows, autonomous navigation, visual global positioning, and many more. The three methods under consideration, all feature-based reconstruction techniques, are DPR,^{30,31–34} PVR,^{21,22} and VSFM.^{13,14,20,56} As this comparison section will show, our feature-based reconstruction method is the fastest of the three methods and produces the densest point cloud. However, it contains noise and is prone to a layer effect due to limited resolution as mentioned in earlier sections. The PVR, otherwise called voxel-based modeling, produces the most complete reconstruction and is quite accurate (visually); however, it is computationally expensive and requires occasional user interference to produce its visually appealing models. The VSFM method is also a feature based method and is compatible with a variety of data types. An additional layer of complexity in evaluating the methods is that each has a different output as shown in Fig. 28. Our methodology, DPR, focuses on producing a dense, accurate point-cloud model. PVR computes an SIFT-based point cloud and continues to build a volumetric scene using Bayesian estimations of intensities, reflections, etc. Finally, VSFM also produces an SIFT-based point cloud and continues with surface refinement techniques to make the model more visually appealing.

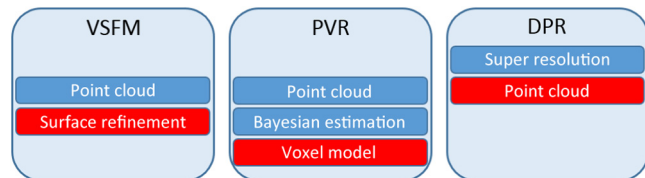


Fig. 28 Comparison of stages in visual structure from motion (VSFM), probabilistic volumetric representation (PVR), and DPR. The most visually appealing output for each technique is marked in red; however, the point-cloud algorithmic step is embedded within all three techniques.

7.1 Experimental Description

As 3-D reconstruction is still an emerging research field, no standardized dataset for reconstruction has been used in publications. On the contrary, each technique designed its algorithms with a specific application in mind and displayed its results of local environments. Our reconstruction algorithm is written in C/C++ and leverages with several optimized functionalities of OpenCV. In order to get a fair assessment of the capabilities of each technology, we conduct a series of tests to create adaptive criteria with which we evaluate. Our first comparison treats each technique as a complete system. We evaluate the visual appeal of the final output of each technique. Although this metric is subjective, similar to image enhancement, visual assessment of 3-D models is also one of the most used evaluation methods and the simplest to understand. This evaluation is also useful for certain applications that utilize 3-D models to create an RW feel. Our second comparison deals with metric values of the model. Since the only stage present in three techniques is the point-cloud algorithmic stage, we compare the densities of the point-cloud stages. This evaluation provides us with valuable insight, since each technique heavily relies on the points within the point-cloud stage to proceed with further processing. The more points a model contains, the more information is available for further algorithmic stages. The final comparison deals with the computational expense associated with each technique. Since these three techniques rival one another in potential deployment onto a real-time system,

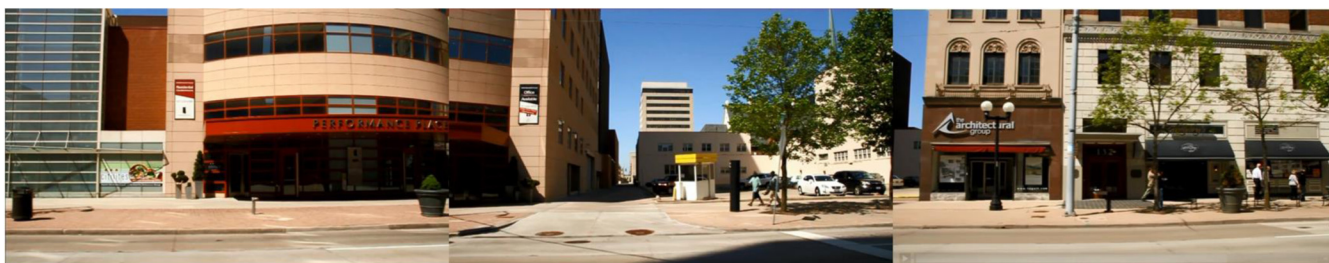


Fig. 29 Frames from the low altitude downtown Dayton dataset.

it is important to consider the computational speed and the ability to accelerate the algorithms.

Since each of these techniques has been developed and modified based on evaluations of a specific dataset, it is natural for each technique to perform at its best on its own dataset. For our evaluation, we consider three datasets that represent a variety of altitudes. Hence, the pixel per inch of each environment is different as well as the change in view point. Our first dataset is a low-altitude (~ 5 feet) street-level environment of downtown Dayton, in which buildings, trees, and sky are visible in the scene. This is an important scenario because some of the techniques are not able to handle objects infinitely far from the camera plane. These techniques are designed for aerial imagery that always contains the ground in view. The second dataset is a medium-altitude (~ 500 feet) set of Providence, Rhode Island, with fairly consistent altitude throughout. This is the dataset preferred by PVR and has been used to illustrate impressive models. The third and final dataset is of high-altitude (~ 7000 feet) CLIF data captured of Columbus, Ohio. This dataset is captured at 7000+ feet with a frame rate of 2 frames per second. We will show a comparison of the three techniques across all three datasets.

7.2 Comparison, Metrics, and Analysis

We will begin with the visual evaluation of the resulting models. Note that each technique produces a different output

type; therefore, this evaluation will be strictly based on visual appeal. Furthermore, we present the metrics that correspond to the density of the point-cloud reconstruction and computational expenses associated with each technique.

7.2.1 Visual evaluation

We present this evaluation in a series of figures. In each figure, we illustrate the output model from each technique. We go on to analyze the resulting models and highlight the advantages and disadvantages of each technique. We begin by evaluating the three techniques on low-altitude imagery. We use the downtown Dayton, Ohio, dataset. Several frames from the scene are depicted in Fig. 29.

These data are recorded in extremely low altitude from a vehicle traveling ~ 25 mph and recorded at an angle perpendicular to the motion of the vehicle. This experiment best simulates a low flying UAS or a UGV performing reconstruction at ground level. Notice the downtown Dayton scene contains roads, buildings, and trees as well as sky image regions. In Fig. 30, we illustrate the resulting point clouds. We begin by analyzing the reconstruction by VSFM. As Fig. 30(a) indicates, the scene model is well defined and correctly colored. It does not contain as many points as DPR. The surface refinement appears to create a blurring effect of the actual point locations. Overall, however, this is strong result considering VSFM had no adjustment made to fit the dataset.

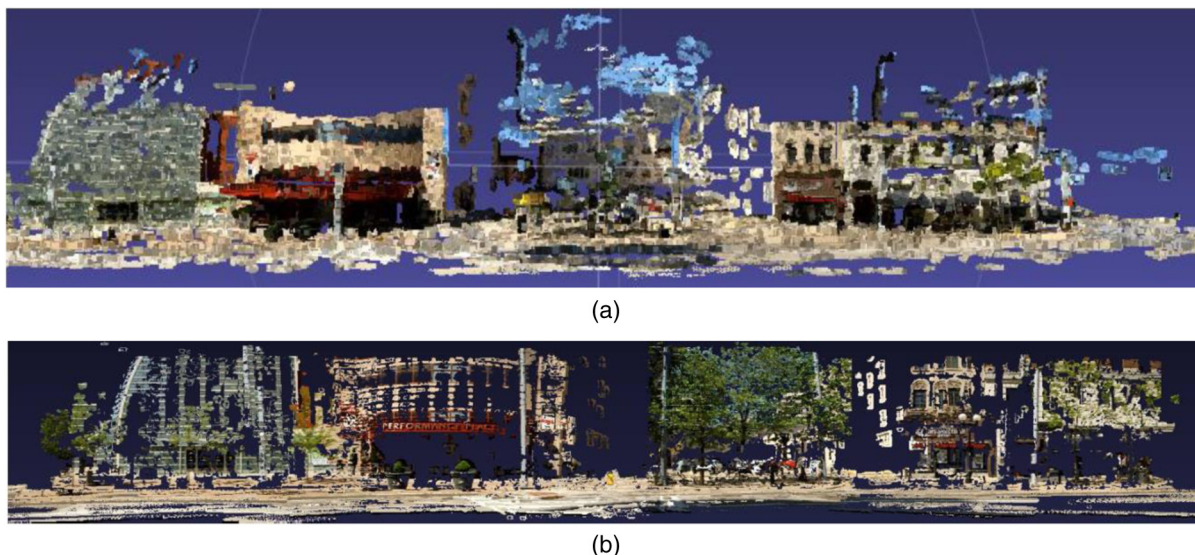


Fig. 30 The 3-D reconstruction results using the low altitude (Dayton, Ohio) dataset: (a) VSFM results and (b) DPR results.

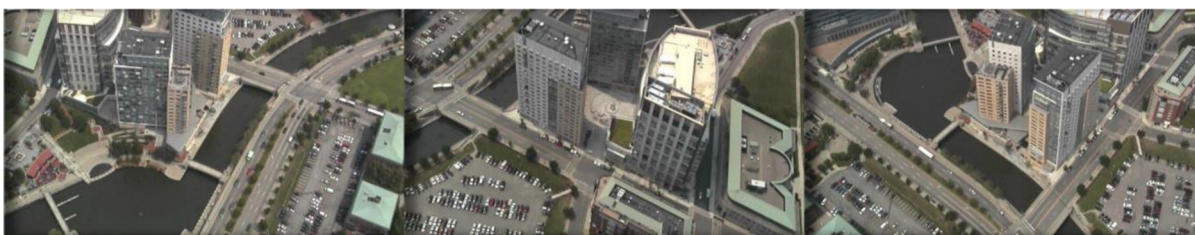


Fig. 31 Frames from the medium altitude Providence, Rhode Island, dataset.

Next, we observe the reconstruction from DPR in Fig. 30(b). As shown in previous section, the model is dense and accurately represents the scene. The density of the model allows us to read labels on signs and buildings. Overall, this comes as no surprise that the technique outperforms other algorithms on its prime dataset. It is important to note that the PVR representation is not able to construct a model due to the presence of infinitely distant objects in the scene, such as sky. Due to the scene complexity, the algorithm was not able to properly assign the initial voxel grid.

The second scenario depicts a medium-altitude scene of Providence, Rhode Island. In this scene, a helicopter circles downtown Providence several times at a varying speed, viewing angle, and altitude. Figure 31 illustrates several frames from the video sequence. We begin by evaluating the VSFM result in Fig. 32(a). The first observation that sticks out is the sharpness of the edges where vertical and horizontal planes meet. The buildings and the ground form almost a right angle. Although the model does not appear dense, the smearing that was evident in the previous scenario is not obvious in the medium-altitude case. Overall, VSFM again produced an acceptable model that can be enhanced further. We continue the evaluation by analyzing the model generated by PVR in Fig. 32(b). Visually, this model surpasses the others completely. The model does not appear as a point cloud but rather as a rendered 3-D scene. It is

comparable to manually created video game 3-D environments. There exists the occasional blurriness, but what makes PVR so appealing is the constant Bayesian estimation updates. The scene continues to improve as more frames are added. Overall, the PVR produced the best visually appealing results for the medium-altitude dataset; however, that comes with little surprise as the dataset is provided by the authors. PVR is able to generate visually appealing results on the several datasets provided by the author; however, we found that on many other datasets, the algorithm fails. Our final evaluation for the medium-altitude data is of DPR in Fig. 32(c). Immediately, we notice the elevation differences between the parking lot and the skyscrapers. Interestingly, the layer effect happens at an angle in this scenario. Because this dataset is captured at an angle in relation to the ground as well as in a circular motion, we see the layer effect produced stripes across the buildings. Also, the algorithm utilizes piece-wise linear motion to compute its disparities and depths, causing more prominent noise points. Overall, the DPR by visual inspection produces a dense, accurate, and noise-prone representation of the scene in medium altitudes.

Our final dataset is the high-altitude CLIF over Columbus, Ohio. This dataset poses unique problems because of the high altitude at which images are captured. Objects of interest in the scene appear as low-resolution objects. We begin

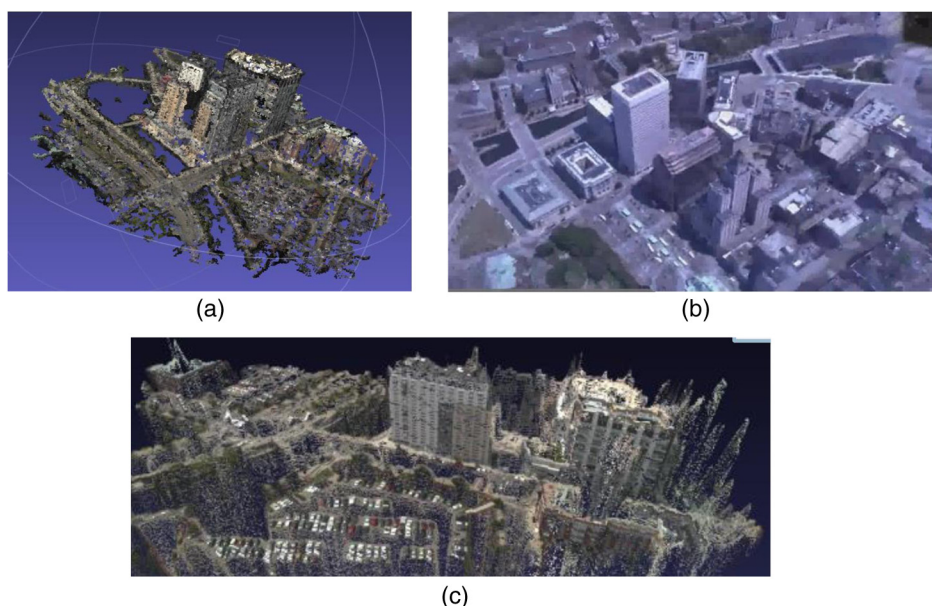


Fig. 32 The 3-D reconstruction results using the medium altitude (Providence, Rhode Island) dataset: (a) VSFM results, (b) PVR results, and (c) DPR results.

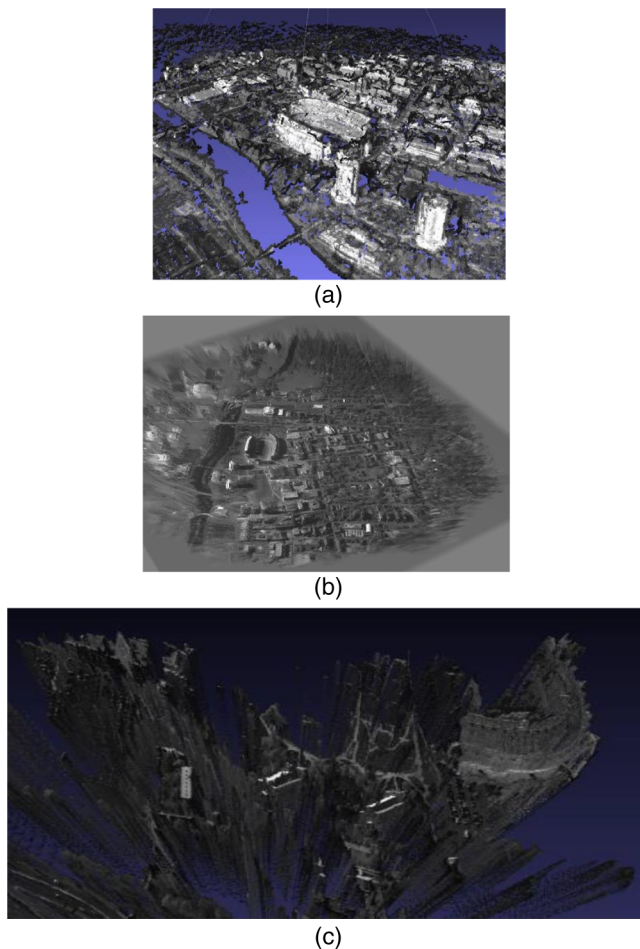


Fig. 33 The 3-D reconstruction results using the high altitude (Columbus, Ohio) dataset: (a) VSFM results, (b) PVR results, and (c) DPR results.

by evaluating the performance of VSFM in Fig. 33(a). The results indicate another sparse and accurate point-cloud representation. The technique is able to handle the low resolution of buildings to create sharp edges with the ground plane. The infamous Ohio State “horse shoe stable” sticks out above the rest in the 3-D model. Overall, VSFM again produced an acceptable model that can contain sparse points blurred to take up more of the scene. In Fig. 33(b), we observe the results produced by PVR. Similar to the medium-altitude dataset, this technique produces a visually

appealing model. The model does not contain the break and hole seen in a point-cloud representation. The edges of the model are extremely blurred due to the lack of imagery that covers those areas. As new imagery is processed, the model is continuously updated. Overall, PVR is the most visually appealing model when considering high-altitude imagery. It is important to note that the CLIF data used for this reconstruction were acquired from the PVR data repository; hence, the technique was adjusted and trained to handle that particular dataset. Our final visual evaluation is of DPR on the CLIF data. Unfortunately, this dataset did not produce great reconstruction results. Due to the low frame rate and low resolution, the optical flow features were unable to be properly matched and the further mismatched points dominate the model in Fig. 33(c). Overall, DPR relies on high resolution and high disparity resolution, and the high-altitude CLIF dataset was not enough to produce impressive and accurate results. Figure 34 provides a summary of the visual assessment. ³³ The ratings represent the median score taken from a survey given to field experts. The figure contains the visual assessment rating for each scenario, where a 3 indicates excellent and 1 indicates very poor or nonexistent.

7.2.2 Density and computational expense metrics

In order to get a better sense of the capabilities of each of the three techniques, we begin by evaluating the number of points each technique produces. Table 5 summarizes the metrics of the comparison. ³³ It is broken into three separate tables, where each reconstruction technique is evaluated separately. For each technique, we separate the three datasets by row. The first column indicates the number of frames used in the reconstruction and their size in column four. This is important to note, since with more frames more information becomes available about the scene. Next, we present the time it takes to process and compute the model on a dual core HP Z600 machine with 8 GB RAM and an Nvidia GeForce GTX 980 4 GB video card. These time values vary from seconds of computations to hours. We also present the number of points that compose each model.

From the table, we learn that the computation time is extremely different for each reconstruction technique. While PVR takes on the order of hours to process and produce a 3-D model, VSMF works on the order of several minutes, and DPR completes its point-cloud representation on the order of seconds. These time differences reflect the fundamental differences in the approach to the reconstruction

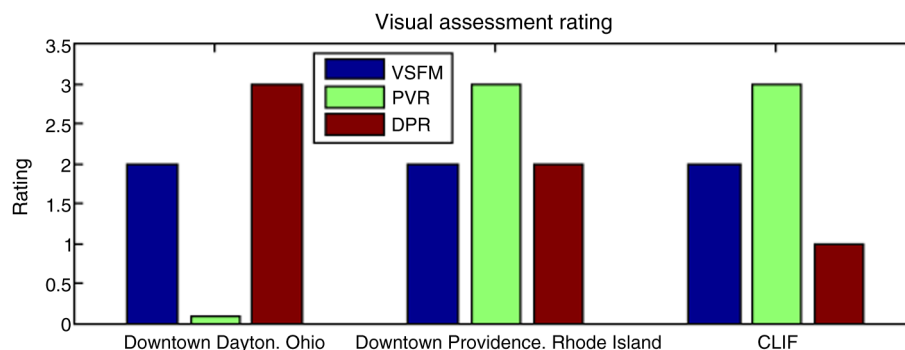


Fig. 34 A summary of the visual assessment ratings. ³³

Table 5 Analysis of metrics for each evaluation technique.³³

	Dataset	No. of frames	Time to compute	No. of points	Frame size
Visual structure from motion ¹⁴	Downtown Dayton, Ohio	133	21 min	303,403	1280 × 720
	Downtown Providence, Rhode Island	112	16 min	115,824	1280 × 720
	CLIF	101	30 min	820,026	2672 × 2004
Probabilistic volumetric representation (PVR) ^{21,22}	Downtown Dayton, Ohio	133	DNC ^a	458,351	1280 × 720
	Downtown Providence, Rhode Island	112	2.5 h	268,269	1280 × 720
	CLIF	101	3.5 h	1,257,691	2672 × 2004
Dense point-cloud representation	Downtown Dayton, Ohio	133	43 s	2,632,834	1280 × 720
	Downtown Providence, Rhode Island	112	8 s	802,501	1280 × 720
	CLIF	101	125 s	2,654,141	2672 × 2004

CLIF, Columbus Large Image Format.

^aDNC (did not compute) is given when a technique is unable to generate a 3-D model. PVR was not able to initialize due to image regions infinitely far from the camera plane, such as sky.

solution. While additional time is spent on the beautification of the model in PVR, the usability for our application does not increase. Therefore, we conclude that the technique that generates a point cloud the quickest and most accurately is the optimal technique. When comparing the three techniques, our DPR is magnitudes of time faster in computation than VSFM and PVR. The complexity of the feature matching and description in VSFM is an expensive function. At the same time, PVR surface multihypothesis estimation also creates a severe time lag.

We continue our evaluation of the techniques by comparing the number of points present in each model. While PVR does not appear as points, it utilizes SIFT feature to construct the environment. The number of points associated with PVR is the number of features extracted for their computations. When considering the number of points, DPR is again much denser than VSFM and PVR. This density allows the viewer to identify and read signs and labels in the scene. Our final comparison evaluates the point-cloud stage of each technique.

7.2.3 Point-cloud evaluation

We have already compared the visual output of each model, the density of the model, and the computational expense. We evaluate the usability of the model by observing the point-cloud representation. As mentioned throughout this paper, our application focuses on using 3-D models for determining occlusions, elevation changes, scene changes, aid in autonomous navigation, and many more. We will describe the type of model needed for these applications.

When considering autonomous navigation, a UGV or UAS would need to know where obstacles are located in order to navigate around them. Similarly, when considering an occlusion detector, the system relies on firmly knowing where there is an object and where there is not. Therefore, to assess the usability of the 3-D reconstruction system, we must remove the beautification stages and bring them to

a thresholded true point cloud. Then algorithms such as ray tracing^{3,57} can be applied to determine occlusions and obstacles in the scene. Out of the three reconstruction techniques, DPR and VSFM already output those concrete point-cloud values. PVR, however, relies on multiple hypotheses for surface estimation and, therefore, does not contain concrete points until a thresholding has taken place. In Fig. 35, we present the concrete point-cloud models generated by each of the three techniques that would be used for ray tracking.³³

It can be observed that in pure thresholded form, VSFM and PVR appear more noisy than in previous demonstrations. In Fig. 35, we evaluate the point clouds generated using the video captured over downtown Providence, Rhode Island. From Fig. 35(a), we determine that VSFM appears relatively accurate with distinct sharp corners and edges of buildings. The PVR point cloud, Fig. 35(b), appears to be the most noisy. This is due to the thresholding done on the voxel model. By thresholding, voxels with a surface characteristic above a certain value are labeled as points. Therefore, points can appear anywhere in the model irrespective of their relative location to other points. As a result, the point cloud indicates points appearing above the buildings and in the sky regions. In Fig. 35(c), we present the DPR point-cloud model. This image also displays the sharp edges and corners formed at the base of the buildings. As presented throughout this section, we can conclude that DPR provides the most dense and most usable point-cloud model for autonomous navigation with the quickest computational timing.

The traveling speed and frame rate of the camera affect the baseline between each pair of images and, thus, the resulting model. High speeds need to be compensated with higher frame rates in order to provide enough frames of the scene for points to be matched and triangulated into the 3-D model. For the most part, the feature matching will not be affected by speed/frame rate alterations; however, extremely

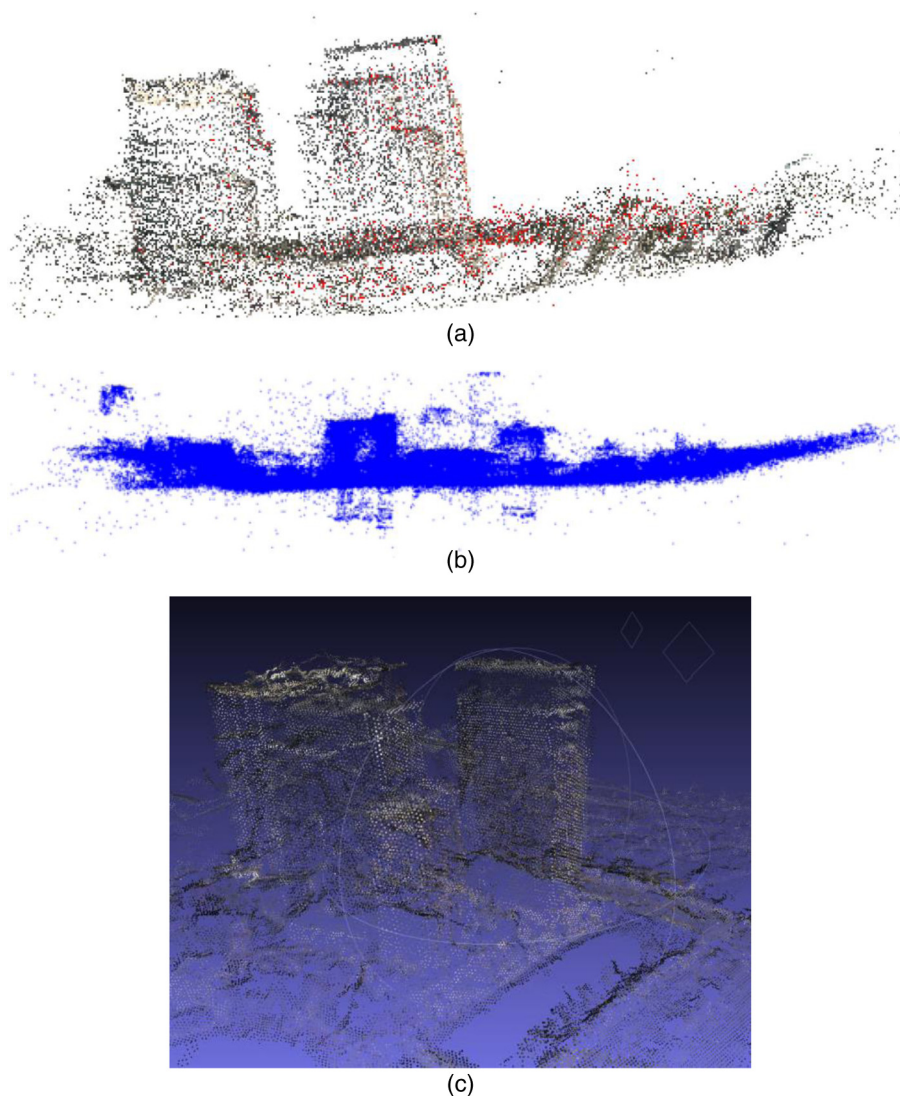


Fig. 35 The thresholded models of the three techniques illustrate the model as they would be used in application: (a) VSFM, (b) PVR, and (c) DPR. Results are generated using our implementations of the algorithm.³³

high speeds can also create more motion blur within the images and that can cause more noise in the model.

8 Conclusion

In this paper, we have presented a novel 3-D reconstruction algorithm. Our technique focuses on reducing the computational expense and enhancing the density, usability, and visual appeal over the two presented state-of-the-art techniques. By adding optical flow features in the algorithmic framework, we were able to create a dense 3-D point-cloud model. As the results indicated, the new point clouds allow the user to distinguish the scene by clearly identifying buildings, cars, and trees. With the introduction of a denser point cloud, we also introduce the layer effect, in which numerous points contain the same depth coordinate. We have presented and evaluated a super-resolution technique that is designed to enhance the depth resolution of the resulting point-cloud models, thus negating the layer effect. After assessing the effects of super resolution on various types of imagery, we concluded that by including super resolution into our

algorithmic architecture, we would increase the density (number of points tracked) and the depth resolution via an increase in the number of discrete disparities (increase the number of layers). However, the noise points that, in the prior model, were barely noticeable have also been enhanced due to super resolution. On the postprocessing end, we identify the commonality in the noise speckles within the point-cloud model and use dynamic filters to suppress that noise in the model. The presentation of the framework, optical flow, super resolution, and noise suppression concludes our presentation of the DPR technique.

We compare the DPR models to other representation techniques presented, VSFM and PVR. The current state-of-the-art PVR has proven to create appealing 3-D models. We have also evaluated an accurate feature based modeling technique, VSFM. We showed that when compared to our DPR reconstruction technique, VSFM and PVR fall short in the density comparison, in the computational expense comparison, and in the usability comparison. We have proven through metrics and analysis that the contributions of DPR

will, in the future, lead to the development of a successful and deployable reconstruction system.

As described in this paper, the 3-D reconstruction field does not contain a standardized dataset with ground truth information. Each technique provides 3-D model results on data capture in their proximity. Objective metrics include point-cloud density and computation time. Visual appeal is subjective; however, with a sufficiently large polling, some conclusions can be drawn on the model. In conclusion, at its current state, the algorithm does not contain the computations needed to handle movement within the scene. It is envisaged that with a sufficiently high frame rate and camera speed, moving objects can be reconstructed using this algorithm as is, because the moving objects would appear stationary in the imagery. In the future, we plan to compare our computed 3-D models to models acquired with a laser ranger finder to prove ground truth.

References

1. Y. Furukawa et al., "Reconstructing building interiors from images," in *IEEE 12th Int. Conf. on Computer Vision*, Kyoto, pp. 80–87, IEEE (2009).
2. D. Gallup et al., "Real-time plane-sweeping stereo with multiple sweeping directions," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, pp. 1–8, IEEE (2007).
3. S. Kashyap et al., "Real time ray tracing of point-based models," in *Proc. of the ACM SIGGRAPH Symp. on Interactive 3D Graphics and Games: Posters*, p. 4, ACM, New York (2010).
4. D. Lee and J. Park, "Estimation of camera parameters from a single moving camera using quaternion-based interpolation of 3D trajectory," in *Computer Graphics, Imaging and Visualisation*, Bangkok, pp. 77–80, IEEE (2007).
5. B. Peasley and S. Birchfield, "Replacing projective data association with Lucas-Kanade for KinectFusion," in *IEEE Int. Conf. on Robotics and Automation*, Karlsruhe, pp. 638–645, IEEE (2013).
6. A. J. Davison and D. W. Murray, "Simultaneous localization and map-building using active vision," *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(7), 865–880 (2002).
7. D. Di Paola et al., "An autonomous mobile robotic system for surveillance of indoor environments," *Int. J. Adv. Robot. Syst.* **7**(1), 19–26 (2010).
8. A. Flint et al., "Growing semantically meaningful models for visual slam," in *IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, California, pp. 467–474, IEEE (2010).
9. R. Fabbri and B. Kimia, "3D curve sketch: flexible curve-based stereo reconstruction and calibration," in *IEEE Conf. on Computer Vision and Pattern Recognition*, San Francisco, California, pp. 1538–1545, IEEE (2010).
10. C. L. Fennema and W. B. Thompson, "Velocity determination in scenes containing several moving objects," *Comput. Graph. Image Process.* **9**(4), 301–315 (1979).
11. H. Ishikawa and D. Geiger, "Occlusions, discontinuities, and epipolar lines in stereo," in *European Conf. on Computer Vision*, pp. 232–248, Springer (1998).
12. S. Agarwal et al., "Building Rome in a day," in *IEEE 12th Int. Conf. on Computer Vision*, pp. 105–112, ACM, New York (2011).
13. C. Wu et al., "Multicore bundle adjustment," in *IEEE Conf. Computer Vision and Pattern Recognition*, Providence, Rhode Island, pp. 3057–3064, IEEE (2011).
14. C. Wu, "VisualSFM: a visual structure from motion system," 2011, <http://ccwu.me/vsfm/> (5 April 2013).
15. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.* **60**(2), 91–110 (2004).
16. D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. of the Seventh IEEE Int. Conf. on Computer Vision*, Kerkyra, Vol. 2, pp. 1150–1157, IEEE (1999).
17. B. Triggs et al., "Bundle adjustment—a modern synthesis," in *Vision Algorithms: Theory and Practice*, B. Triggs, A. Zisserman, and R. Szeliski, Eds., pp. 298–372, Springer (2000).
18. S. Birchfield and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," in *Sixth Int. Conf. on Computer Vision*, pp. 1073–1080, Kluwer Academic Publishers (1998).
19. R. Szeliski and P. Golland, "Stereo matching with transparency and matting," in *Sixth Int. Conf. on Computer Vision*, Bombay, pp. 517–524, IEEE (1998).
20. C. Wu et al., "3D model matching with viewpoint-invariant patches (VIP)," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Anchorage, Alaska, pp. 1–8, IEEE (2008).
21. T. Pollard and J. L. Mundy, "Change detection in a 3-D world," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, pp. 1–6, IEEE (2007).
22. M. I. Restrepo, B. A. Mayer, and J. L. Mundy, "Object recognition in probabilistic 3D volumetric scenes," in *Proc. of Int. Conf. on Pattern Recognition Applications and Methods*, Vilamoura, Algarve, Portugal, Vol. 1, pp. 180–190 (2012).
23. M. Agrawal and L. S. Davis, "A probabilistic framework for surface reconstruction from multiple images," in *Proc. of the 2001 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, Vol. 2, p. II-470, IEEE (2001).
24. N. Cornelis et al., "3D urban scene modeling integrating recognition and reconstruction," *Int. J. Comput. Vis.* **78**(2–3), 121–141 (2008).
25. A. Golovinskiy, V. G. Kim, and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments," in *IEEE 12th Int. Conf. on Computer Vision*, Kyoto, pp. 2154–2161, IEEE (2009).
26. X. Li et al., "Modeling and recognition of landmark image collections using iconic scene graphs," in *European Conf. on Computer Vision*, pp. 427–440, Springer (2008).
27. Y. Diskin et al., "UAS exploitation by 3D reconstruction using monocular vision," in *24th Int. Technical Meeting of the Satellite Division of the Institute of Navigation-ION GNSS*, pp. 3596–3604, Oregon Convention Center, Portland, Oregon (2011).
28. H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: speeded up robust features," in *European Conf. on Computer Vision*, pp. 404–417, Springer (2006).
29. R. C. Tompkins et al., "3D reconstruction from a monocular vision system for unmanned ground vehicles," *Proc. SPIE* **8186**, 818608 (2011).
30. Y. Diskin et al., "3D scene reconstruction using monocular vision," presented at *Ohio Innovation Summit: Sensors Field Day*, Dayton, Ohio (16 June 2011).
31. Y. Diskin and V. K. Asari, "Dense point-cloud creation using super resolution for a monocular 3D reconstruction system," *Proc. SPIE* **8399**, 83990N (2012).
32. Y. Diskin and V. K. Asari, "Adaptive noise suppression technique for dense 3D point cloud reconstructions from monocular vision," *Proc. SPIE* **8499**, 84991B (2012).
33. Y. Diskin and V. K. Asari, "3D scene reconstruction for aiding unmanned vehicle navigation," in *Proc. of the 2003 IEEE Int. Conf. on Systems, Man, and Cybernetics*, pp. 243–248, IEEE Computer Society, Washington, DC (2013).
34. Y. Diskin and V. K. Asari, "Dense 3D point-cloud model using optical flow for a monocular reconstruction system," in *IEEE Applied Imagery Pattern Recognition Workshop: Sensing for Control and Augmentation*, Washington, DC, pp. 1–6, IEEE (2013).
35. K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in *European Conf. on Computer Vision*, Vol. 2350, pp. 128–142, Springer (2002).
36. Y. Furukawa et al., "Manhattan-world stereo," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Miami, Florida, pp. 1422–1429, IEEE (2009).
37. K. Konolige, "Stereo Geometry," SRI International, August 1999, <http://publ.willowgarage.com/~konolige/svs/disparity.htm> (11 January 2011).
38. K. Nakayama et al., "Optical velocity patterns, velocity-sensitive neurons, and space perception: a hypothesis," *Perception* **3**(1), 63–80 (1974).
39. A. Saxena, M. Sun, and A. Y. Ng, "Make3D: learning 3D scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(5), 824–840 (2009).
40. J. R. Seal, D. G. Bailey, and G. S. Gupta, "Depth perception with a single camera," in *Proc. of Int. Conf. on Sensing Technology*, pp. 96–101, Massey University, Palmerston North, New Zealand (2005).
41. A. Schulz et al., "CUDA SURF—a real-time implementation for SURF," 2011, <https://www.d2.mpi-inf.mpg.de/surf?q=surf> (17 December 2014).
42. B. K. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.* **17**(1), 185–203 (1981).
43. B. K. Horn and E. Weldon Jr., "Direct methods for recovering motion," *Int. J. Comput. Vis.* **2**(1), 51–76 (1988).
44. M. Islam et al., "A kernel regression based resolution enhancement technique for low resolution video," in *Digest of Technical Papers Int. Conf. on Consumer Electronics*, Las Vegas, Nevada, pp. 29–30, IEEE (2010).
45. M. M. Islam et al., "Super-resolution enhancement technique for low resolution video," *IEEE Trans. Consum. Electron.* **56**(2), 919–924 (2010).
46. H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Proc. of the 2004 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, Washington, DC, Vol. 1, p. I-275, IEEE (2004).
47. D. Glasner, S. Bagon, and M. Irani, "Super-resolution from a single image," in *IEEE 12th Int. Conf. on Computer Vision*, Kyoto, pp. 349–356, IEEE (2009).
48. K. S. Ni et al., "Single image superresolution based on support vector regression," in *Proc. of 2006 IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Toulouse, Vol. 2, pp. 601–604, IEEE (2006).

49. E. Meijering, "A chronology of interpolation: from ancient astronomy to modern signal and image processing," *Proc. IEEE* **90**(3), 319–342 (2002).
50. R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Trans. Acoust. Speech Signal Process.* **29**(6), 1153–1160 (1981).
51. C. Harris and M. Stephens, "A combined corner and edge detector," 1988, <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.231.1604> (10 February 2015).
52. P. F. Maenner, "Emerging standards suite for wide-area ISR," *Proc. SPIE* **8386**, 83860J (2012).
53. AFRL, "Columbus large image format," 2006, <https://www.sdms.afrl.af.mil/index.php?collection=clif2006> (1 October 2012).
54. J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**, 679–698 (1986).
55. D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc. Lond. B Biol. Sci.* **207**(1167), 187–217 (1980).
56. C. Wu et al., "Schematic surface reconstruction," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Providence, Rhode Island, pp. 1498–1505, IEEE (2012).
57. L. Linsen, K. Müller, and P. Rosenthal, "Splat-based ray tracing of point clouds," 2007, <https://dspace.zcu.cz/handle/11025/1426> (10 February 2015).

Yakov Diskin is a PhD candidate in electrical and computer engineering at the University of Dayton, Dayton, Ohio. Specializing in computer vision, his primary research focus is in the field of object tracking and depth estimation in wide-area motion imagery, where he has completed 37 publications in the related areas. As an active contributor to the University of Dayton Vision Lab, his research interests include biometrics, unmanned-aerial-systems, and real-time image enhancement.

Vijayan Asari is a professor in electrical and computer engineering and Ohio Research Scholars Endowed Chair in wide-area surveillance at the University of Dayton, Dayton, Ohio. He is the director of the University of Dayton Vision Lab (Center of Excellence for Computer Vision and Wide Area Surveillance Research). He holds three patents and has published more than 450 research papers, including 79 peer-reviewed journal papers. He is a senior member of IEEE and SPIE.